AFIT/GE/ENG/97D-19

Cepstral Processing For GPS Multipath Detection and Mitigation

THESIS

Charles D. Ormsby
Captain, USAF

AFIT/GE/ENG/97D-19

19980128 100

Approved for public release; distribution unlimited

DTIC QUALITY INSPECTED 3

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

AFIT/GE/ENG/97D-19

Cepstral Processing For GPS Multipath Detection and Mitigation

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

Charles D. Ormsby, B.S.
Captain, USAF

December, 1997

Approved for public release; distribution unlimited

AFIT/GE/ENG/97D-19

Cepstral Processing For GPS Multipath Detection and Mitigation

Charles D. Ormsby, B.S.

Captain, USAF

Approved:

Capt. Stewart DeVilbiss Ph. D. Thesis Advisor	3 Dec 97 Date
M. Pachter Dr. Meir Pachter Committee Member	3 Dec 97 Date
Maj. Michael Temple Ph. D. Committee Member	3 Dec 97 Date

Acknowledgements

The author would like to thank the members of the faculty committee who reviewed this thesis, Dr. Michael Temple and Dr. Meir Pachter. Thank you for the time and energy you put into reading and critiquing this work, and for the helpful pointers along the way. Special thanks goes to the advisor, Dr. Stewart DeVilbiss. Thank you for the freedom to attack this project in my own way, and for the many hours spent reviewing drafts of this work. Thanks for your conscientious efforts to cross every t and dot every i, both technically and grammatically. Your help greatly improved the quality of this thesis. Finally, the author wishes to thank the students of the December 1997 engineering class at the Air Force Institute of Technology, and particularly Fred Baier. Countless hours were spent working together as a team to ensure we all gained the knowledge necessary to prepare us for our thesis research and beyond. The team spirit at AFIT has made it a joy to learn here, thanks to all.

Charles D. Ormsby

Table of Contents

Page

Acknowledgements iii

List of Figures viii

List of Tables xi

List of Abbreviations xii

List of Symbols xiii

Abstract xiv

I. Introduction 1

- 1.1 Overview 1
- 1.2 GPS Background 1
  - 1.2.1 The GPS Signal 2
  - 1.2.2 Spreading Code 2
  - 1.2.3 Code Tracking 4
- 1.3 Multipath Interference 5
  - 1.3.1 Code Tracking in the Presence of Multipath 6
- 1.4 Previous Efforts in GPS Multipath Mitigation 6
  - 1.4.1 Narrow Correlator Spacing 7
  - 1.4.2 Multipath Estimating Delay Lock Loop 7
  - 1.4.3 Modified RAKE Delay Lock Loop 7
  - 1.4.4 Correlator Reference Waveform Design 8
- 1.5 Problem Statement 8
- 1.6 Objectives 9
- 1.7 Assumptions 10
- 1.8 Approach 10

Page

II. Cepstral Analysis 11
  2.1 Introduction 11
  2.2 Definitions of the Power and Complex Cepstra 11
    2.2.1 Power Cepstrum Definition 11
    2.2.2 Complex Cepstrum Definition 12
  2.3 Cepstral Processing of Multipath Signals 13
    2.3.1 Power Cepstrum 14
    2.3.2 Power Cepstrum Example 22
    2.3.3 Complex Cepstrum 23
  2.4 Filtering in the Cepstral Domain 27
    2.4.1 Multipath Mitigation Example 28

III. Complex Cepstrum Adaptive Filter 32
  3.1 Overview 32
  3.2 Received Signal Model 32
  3.3 Complex Cepstrum Adaptive Filter 32
    3.3.1 Overview 32
    3.3.2 Zero Padding 33
    3.3.3 Cepstra Calculations 34
    3.3.4 Peak Detection 34
    3.3.5 Adaptive Weighted Comb Filter 35
    3.3.6 Inverse Complex Cepstrum 37
  3.4 The Non-Coherent Delay Lock Loop 37
  3.5 Conclusion 51

IV. Computer Simulations and Results 52
  4.1 Overview 52
  4.2 Simulation Parameters 52

Page

4.2.1 DS/SS Spreading Code 53

4.2.2 Carrier and Direct Path Signals 53

4.2.3 Multipath Parameters 53

4.2.4 NCDLL Design 54

4.2.5 Simulation Parameters Summary 54

4.3 Signal-to-Noise Ratio Analysis 55

4.4 Cepstrum Processing Analysis 59

4.4.1 Multipath Detector 59

4.4.2 Multipath Mitigation 69

4.5 Summary 81

V. Conclusion 83

5.1 Recommendations for Future Research 84

5.2 Final Conclusion 85

Appendix A. Matlab Function Files 86

A.1 Overview 86

A.2 Thesis_Simulator Function File 86

A.3 Expand Function File 88

A.4 Delay Function File 89

A.5 Fpeak Function File 90

A.6 Meanfilt Function File 90

A.7 Scurve Funtion File 91

A.8 Vector_Shift Function File 92

A.9 Trackpoint Function File 93

A.10 Maxfind and Minfind Function Files 94

A.11 Interpolate Function File 95

A.12 Summary 96

Page

Bibliography 97

Vita 98

List of Figures

Figure		Page
1.	Maximal Length Spreading Code Autocorrelation Function	4
2.	The Multipath Signal	5
3.	Signals for Power Cepstrum Example	23
4.	Power Cepstrum for Example Signal	24
5.	Complex Cepstrum Filtering Process	24
6.	Power and Complex Cepstrum Multipath Detection and Mitigation Procedure	29
7.	Signals for Complex Cepstrum Example	30
8.	Complex Cepstrum for Example Signal	31
9.	Recovered Direct Path Signal for Example Signal	31
10.	A Code Tracking Loop Employing the Complex Cepstrum Adaptive Filter	33
11.	Typical Power Cepstrum Magnitude for a GPS Multipath Composite Signal	35
12.	Typical Filtered and Unfiltered Complex Cepstrum for a GPS Multipath Composite Signal	36
13.	The Non-Coherent Delay Lock Loop	38
14.	The Non-Coherent Delay Lock Loop S-Curve for $\Delta = 1$ Chip	44
15.	The Non-Coherent Delay Lock Loop S-Curve for $\Delta = 0.5$ Chip	45
16.	The Non-Coherent Delay Lock Loop Non-Linear Model	48
17.	The Non-Coherent Delay Lock Loop Linear Model	50
18.	Complex Cepstrum of Noisy and Ideal Signal	55
19.	Recovered Signal from a Noisy Complex Cepstrum Process	56
20.	Recovered Signal from a Noisy Complex Cepstrum Process (Expanded Time Scale)	57
21.	SNR Threshold Effect	58
22.	Bandpass Filtering Effect on the Complex Cepstrum	59
23.	Power Cepstrum for a GPS Multipath Signal, SNR $=$ 30 dB, Multipath Amplitude $=$ 0.5, Multipath Delay $=$ 0.5	60

Figure Page

24. Power Cepstrum for a GPS Multipath Signal, SNR = 250 dB, Multipath Amplitude = 0.5, Multipath Delay = 0.5 61

25. Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.2 62

26. Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.4 63

27. Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.6 63

28. Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.8 64

29. Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.2 64

30. Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.4 65

31. Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.6 65

32. Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.8 66

33. Multipath Detector Test Results for SNR = 223 dB and Multipath Amplitude = 0.2 66

34. Multipath Detector Test Results for SNR = 223 dB and Multipath Amplitude = 0.4 67

35. Multipath Detector Test Results for SNR = 223 dB and Multipath Amplitude = 0.6 67

36. Multipath Detector Test Results for SNR = 223 dB and Multipath Amplitude = 0.8 68

37. Non-Coherent Delay Lock Loop Tracking Curves, Multipath Normalized Amplitude = 0.6, Multipath Normalized Delay = 0.65 Chips 70

38. Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.2 70

39. Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.4 71

Figure Page

40. Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.6 71

41. Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.8 72

42. Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.2 73

43. Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.4 74

44. Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.6 74

45. Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.8 75

46. Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.2 75

47. Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.4 76

48. Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.6 76

49. Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.8 77

50. Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.2 77

51. Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.4 78

52. Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.6 78

53. Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.8 79

List of Tables

Table Page

1. GPS Spreading Code Characteristics 3

2. Average Data Summary For 1000 dB SNR 81

3. Average Data Summary For 250 dB SNR 81

4. Average Data Summary For 223 dB SNR 82

List of Abbreviations

Abbreviation		Page
(GPS)	Global Positioning System	1
(DGPS)	Differential Global Positioning System	1
(DS/SS)	Direct Sequence Spread Spectrum	1
(BPSK)	Binary Phase Shift Keying	2
(P-code)	Precision Code	2
Y-Code	Encrypted P-code	2
(C/A)	Coarse/Acquisition	2
(SA)	Selective Availability	2
(CDMA)	Code Division Multiple Access	3
(PRN)	Pseudorandom Noise	3
(NCDLL)	Non-Coherent Delay Lock Loop	7
(SPS)	Standard Positioning System	7
(MEDLL)	Multipath Estimating Delay Lock Loop	7
(MRDLL)	Modified RAKE Delay Lock Loop	7
(SNR)	Signal to Noise Ratio	8
(AFIT)	Air Force Institute of Technology	10
(DTFT)	Discrete Time Fourier Transform	12
(CCAF)	Complex Cepstrum Adaptive Filter	32
(IF)	Intermediate Frequency	32
(AWGN)	Additive White Gaussian Noise	32
(DFT)	Discrete Fourier Transform	33
(DLL)	Delay Lock Loop	37
(VCO)	Voltage Controlled Oscillator	48
(RMS)	Root-Mean-Square	50

List of Symbols

Symbol		Page
L1	1575.42 MHz Carrier Frequency	2
L2	1227.6 MHz Carrier Frequency	2
$R_c(\tau)$	Autocorrelation Function	3
c(t)	DS/SS Spreading Code	3
r(t)	Composite Received Signal	6
s(t)	Direct Path Signal	6
m(t)	Multipath Reflection Signal	6
$x_{pc}[nT]$	Power Cepstrum of x(n)	11
$X(z)$	Z-transform of x(n)	12
$\hat{x}[nT]$	Complex Cepstrum of x(n)	12
a_0	Direct Path Attenuation Factor	32
a_1	Multipath Attenuation Factor	32
θ_0	Direct Path Signal Phase	32
θ_1	Multipath Reflection Signal Phase	32
τ_d	Direct Path Signal Propagation Delay	32
τ_m	Multipath Reflection Propagation Delay	32
ω_0	Carrier Angular Frequency	32
$n(t)$	Additive White Gaussian Noise	32
W	Complex Cepstrum Domain Filter Vector	37
K_1	RF-to-IF Conversion Loss Constant	39
Δ	NCDLL Correlator Spacing	39
ω_{IF}	Intermediate Angular Frequency	39
$S_\Delta(\delta)$	NCDLL S-Curve	42
δ	NCDLL Tracking Error	42
α	Multipath Reflection Normalized Delay in Chips	53

AFIT/GE/ENG/97D-19

Abstract

This work presents a novel approach to code phase multipath mitigation for Global Positioning System (GPS) receivers. It uses the power and complex cepstra for multipath detection and mitigation prior to code phase tracking by a standard non-coherent delay lock loop. Cepstral theory is presented to demonstrate how multipath reflection delays can be detected through the use of the power cepstrum. Filtering can then be performed on the complex cepstrum to remove multipath effects in the cepstral domain. Finally, an inverse complex cepstrum is calculated yielding a theoretically multipath free direct path estimate in the time domain. Simulations are presented to verify the applicability of cepstral techniques to the problem of GPS multipath mitigation. Results show that, under noiseless conditions, cepstral processing prior to code tracking by a standard non-coherent delay lock loop leads to lower code tracking biases than direct tracking of the composite multipath signal by a narrow correlator receiver. An exception to this general rule occurs at a multipath reflection delay of exactly 1.0 chip, relative to the direct path, where the cepstral processing provides no improvement. Additionally, cepstral processing provides little or no improvement over a narrow correlator non-coherent delay lock loop when the multipath delay is small, approximately 0.1 chip or less. However, this weakness is common for other multipath mitigation techniques as well. Finally, this work shows that cepstral processing is highly sensitive to additive white Gaussian noise effects, leading to the conclusion that methods of limiting noise effects must be developed before this technique will be applicable in actual GPS receivers.

Cepstral Processing For GPS Multipath Detection and Mitigation

I. Introduction

1.1 Overview

Satellite navigation systems have ushered in a new era in precision navigation. The Global Positioning System (GPS) in particular provides highly accurate position solutions for both civilian and military applications. This has led to it's use in every arena from personal recreation to highly precise "smart" bombs. However, a primary error source still exists in GPS, ranging errors caused by multipath interference. In fact, multipath signals are the dominant error source for Differential GPS (DGPS). Thus many efforts focus on developing techniques to mitigate multipath errors in DGPS.

1.2 GPS Background

The Global Positioning System is a direct sequence spread spectrum (DS/SS) satellite navigation system consisting of 24 satellites providing navigation information to passive receivers. The system operates using the principle of trilateration. Within the received signal, the satellite transmitter embeds precise timing information. The receiver uses the signal propagation delay to calculate the distance to the satellite using $d = v_p * (t_r - t_t)$ where d is the distance from the receiver to the satellite, v_p is the velocity of propagation (usually assumed to be the speed of light in a vacuum), t_t is the time of transmission, and t_r is the time of receipt.

In addition to transmitting precise timing information, each satellite transmits ephemeris data which provides the satellite's position. From knowledge of the positions of at least four satellites and the range to each satellite, a user position solution is calculated. This still leaves two important questions unanswered. First, how does the receiver distinguish between satellites? Second, how does the receiver determine the precise time of transit for the signal being transmitted from the satellite? Both of these questions will be covered in the sections that follow.

1.2.1 The GPS Signal. Before examining the GPS signal, consider a general direct sequence spread spectrum signal. This signal is represented as

$$s(t) = \sqrt{2P}\cos(\omega_0 t + \theta_c(t) + \theta_d(t)) \tag{1}$$

where $s(t)$ is the transmitted signal, P is the carrier power, ω_0 is the carrier angular frequency, θ_c is the spreading code, θ_d is the data stream. Assuming antipodal binary phase shift keying (BPSK) is used for the spreading code and data modulation, an equivalent form of Equation 1 is given in Equation 2

$$s(t) = \sqrt{2P}c(t)d(t)\cos(\omega_0 t) \tag{2}$$

where $d(t)$ is the data stream and $c(t)$ is the spreading code.

The Global Positioning System consists of satellites transmitting two distinct DS/SS signals. The first signal is called Precision code (P-code). This signal is encrypted, a conversion to Y-Code, before transmission, and is reserved for military users. The P-code signal provides position estimates with accuracy on the order of 10 m in the absence of multipath interference. The second, Coarse/Acquisition (C/A) code, is transmitted for all users. The C/A code signal is subjected to selective availability (SA), a form of intentional signal degradation. With SA in operation, C/A code provides position estimates with accuracy on the order of 100 m in the absence of multipath interference. P-code is transmitted at two carrier frequencies, L1 and L2; whereas C/A code is available only on L1. The properties of the GPS signal are summarized in Table 1 (5).

1.2.2 Spreading Code. The Global Positioning System uses orthogonal Gold codes to spread the signal spectrum. For a discussion of Gold codes see (9, 10). These codes allow GPS

Table 1 GPS Spreading Code Characteristics

Parameter	**C/A Code**	**P Code**
Data Rate	50 Hz	50 Hz
Chip Rate = $1/T_c$	1.023 Mchips/sec	10.23 Mchips/sec
Code Period = N (chips)	1023 Chips (1 ms)	$\approx 6 \times 10^{12}$ Chips (1 week)
Carrier Band Designation	L1 1575.42 MHz	L1, L2 1575.42, 1227.6 MHz respectively

to utilize code division multiple access (CDMA) to distinguish which satellite transmitted a given signal. Due to the orthogonality of the codes, the correlation of the received signal will be essentially zero if the locally generated code is not the same as that of the received signal.

GPS receivers exploit knowledge of the spreading code correlation function to synchronize a locally generated replica of the spreading code to the received spreading code. This synchronization is needed to determine range to the satellites; additionally, this tracking of the spreading code accomplishes despreading of the received signal. The code autocorrelation function, $R_c(\tau)$, is defined as

$$R_c(\tau) = \frac{1}{N T_c} \int_0^{N T_c} c(t)\, c(t + \tau\, T_c)\, dt \tag{3}$$

where τ is the independent (time shift) variable, NT_c is the code period in seconds, and $c(\cdot)$ is the spreading code.

For maximal length pseudorandom noise (PRN) codes, the fundamental period of the code autocorrelation function is (5, 11)

$$R_c(\tau) = \begin{cases} 1 - \tau \left(1 + \frac{1}{N}\right) & |\tau| \leq 1 \\ -\frac{1}{N} & 1 < |\tau| < (N-1) \\ \left[|\tau\,| - (N-1)\right]\left(1 + \frac{1}{N}\right) - \frac{1}{N} & (N-1) \leq |\tau| < N \end{cases} \tag{4}$$

where N is the code period in chips.

Since $N \gg 1$, Equation 4 can be approximated as

$$R_c(\tau) \approx \begin{cases} 1 - |\tau| & |\tau| \leq 1 \\ 0 & elsewhere \end{cases} \tag{5}$$

A plot of the Equation 5 is shown in Figure 1. Although Figure 1 shows only a single peak, the periodicity of the spreading code results in a periodic autocorrelation function. Therefore, the peak of Figure 1 is repeated every N chips.

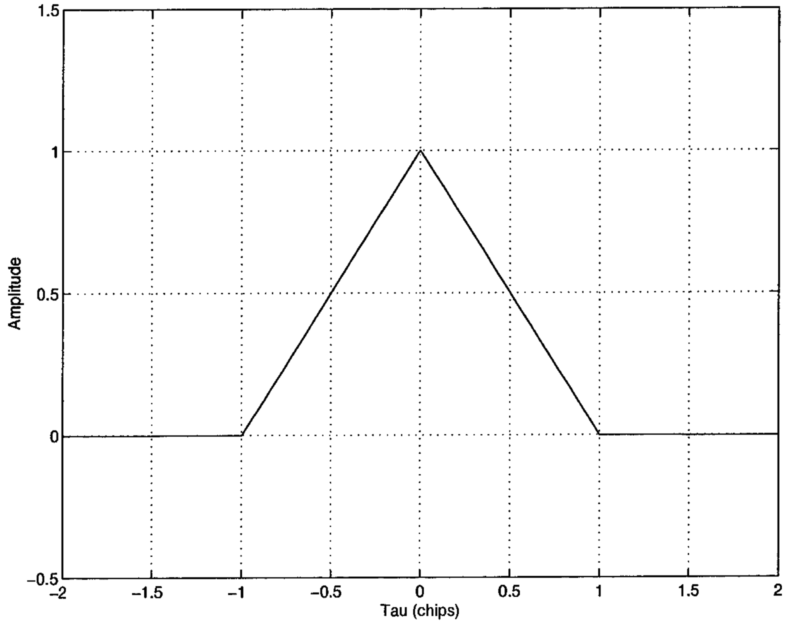

Figure 1 Maximal Length Spreading Code Autocorrelation Function

1.2.3 Code Tracking. As alluded to earlier, code tracking in a DS/SS receiver is accomplished by computing the correlation of the received and locally generated spreading codes. The locally generated code is shifted in time until the correlation is maximized. Maximization of the correlation function occurs at $\tau = 0$. Referring to Equation 3, it is seen that $\tau = 0$ implies that the locally generated code and the received code are synchronized. By measuring the shift of the local code necessary to maximize the correlation function, the receiver can estimate the signal propagation delay. As discussed earlier, the propagation delay is then used to estimate the range to the satellite.

1.3 Multipath Interference

A multipath signal is a reflection of the direct path signal. The multipath phenonmenon is illustrated in Figure 2.

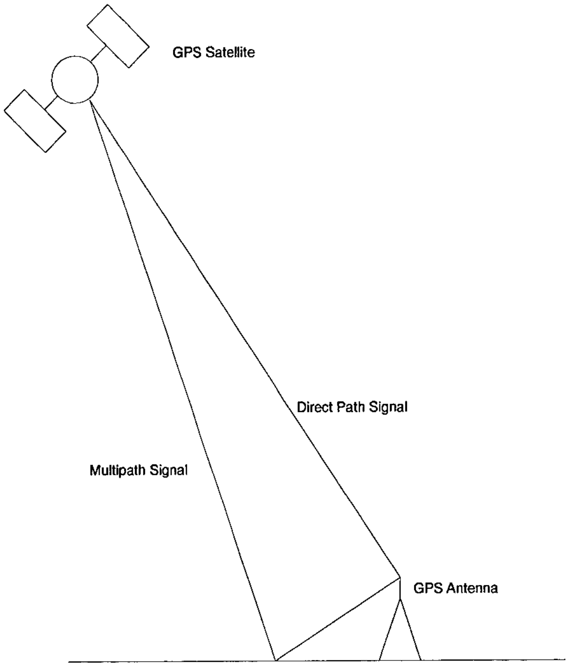

Figure 2 The Multipath Signal

From Figure 2 it is seen that the multipath signal traverses a greater distance than the direct path signal; hence, the multipath signal is a delayed version of the direct path signal. Because it is assumed that the multipath signal is also attenuated with respect to the direct path signal, Equation 2 can be used to represent the i^{th} multipath component as

$$m_i(t) = a_i\, s(t-\tau_{m_i}) = a_i\, c(t-\tau_{m_i})\, d(t-\tau_{m_i})\, \sqrt{2P}\, \cos\left[2\omega_0\, (t-\tau_{m_i})\right] \tag{6}$$

where a_i is the multipath attenuation coefficient and τ_{m_i} is the multipath delay relative to the direct path signal.

Using the superposition of Equations 2 and 6, the total received signal is represented by Equation 7

$$r(t) = s(t) + m(t) = c(t)\, d(t)\, \sqrt{2P}\, \cos\left(\omega_0\, t\right) + \sum_{i=1}^{M} a_i\, c(t - \tau_{m\,i})\, d(t - \tau_{m\,i})\, \sqrt{2P}\, \cos\left[2\omega_0\, (t - \tau_{m\,i})\right] \tag{7}$$

where the summation indicates $M \geq 1$ total multipath reflections.

1.3.1 Code Tracking in the Presence of Multipath. When code tracking in the presence of multipath is performed in the same manner as code tracking in the absence of multipath, the multipath term(s) in Equation 7 cause a distortion of the autocorrelation function. This degradation, which will be discussed later, leads to a non-zero tracking bias with respect to the direct path signal. In other words, rather than tracking the direct path signal perfectly, the multipath components cause some tracking error in the receiver. These effects will be discussed more thoroughly in Chapter III.

1.4 Previous Efforts in GPS Multipath Mitigation

Efforts to eliminate multipath errors in GPS receivers have focused on three main areas: pre-receiver, receiver-internal, and post-receiver. Pre-receiver techniques typically include antenna designs that minimize the antenna gain in the expected direction of the multipath signal. Alternatively, the antenna can be designed to have a physical barrier, such as a ground plane, blocking the multipath signal. Such antenna designs work well in static applications where the trajectory of the multipath signal is relatively constant and can be estimated. However, they do not perform well in dynamic situations where the multipath environment is changing rapidly. Receiver-internal techniques typically consist of digital signal processing techniques designed to eliminate the multipath components from the received signal, or to minimize the negative effects of the multipath components. These techniques perform well even in highly dynamic multipath environments. This thesis fits into the receiver-internal category. A brief summary of other receiver-internal design and signal processing techniques previously developed follows.

1.4.1 Narrow Correlator Spacing. The use of a non-coherent delay lock loop (NCDLL) with a more narrow correlator spacing was one of the first receiver designs for multipath mitigation (12). This technique employs a code tracking delay lock loop with correlators spaced 0.1 of a chip duration apart in time rather than the standard 1 chip width. The narrow spacing results in the delay lock loop tracking a more narrow portion of the correlation peak. This leads to better performance in a multipath environment with maximum tracking errors being reduced by a factor of approximately 10 with respect to the standard correlator structure. This technique is currently used in many production GPS receivers for standard positioning system (SPS) (i. e. C/A code) use.

1.4.2 Multipath Estimating Delay Lock Loop. The multipath estimating delay lock loop (MEDLL) provides improvements over the narrow correlator spacing receiver (13). In the MEDLL, the multipath signal amplitudes and delays are estimated, and the multipath signal is removed through digital signal processing. The MEDLL is implemented using a bank of correlators, each delayed relative to the other by a fraction of a chip. By correlating the received signal using this bank, a sampled version of the distorted correlation function is generated. The MEDLL then uses a maximum likelihood estimator to estimate the multipath signal parameters. Finally, using digital signal processing, the estimated multipath components are removed from the received signal. This proprietary technique provides significant improvement over narrow correlator spacing and is implemented in some production GPS receivers.

1.4.3 Modified RAKE Delay Lock Loop. The modified RAKE delay lock loop (MRDLL) was proposed in an Air Force Institute of Technology thesis by Mark Laxton (5). The MRDLL has similarity to the MEDLL, using a bank of correlators and a maximum likelihood estimation unit to estimate the multipath parameters. The MRDLL employs an adaptive loop controller to adjust the loop filter gain, maintaining a fixed linearized loop natural frequency and damping ratio. This technique has a non-zero tracking error in the absence of multipath interference.

1.4.4 Correlator Reference Waveform Design. Correlator reference waveform design is one of the latest techniques to be suggested for GPS multipath mitigation (14). This technique recommends correlating the received signal with a waveform designed explicitly to reject multipath signals. Rather than using a replica of the GPS spreading code for correlation in the delay lock loop, Weill recommends using the second or fourth derivative of the spreading waveform. This technique claims to reduce multipath ranging errors by increasing the range resolution of the correlation process through the use of novel reference waveforms. The advantage of the second and fourth derivative correlators is that they can be implemented using two correlators, one for code tracking, and one for acquisition, data recovery, and data removal. This is opposed to a bank of many correlators for the MEDLL and the MRDLL. Additionally, this technique can be implemented easily without significant additional signal processing, such as that required by the MEDLL and the MRDLL. The disadvantages are that this technique requires a slightly higher input signal to noise ratio (SNR) than the other techniques, and this technique cannot completely remove the multipath interference. Rather, correlator reference waveform design greatly reduces the error caused by the multipath signals. Like the MRDLL, this technique is very new, and not likely to have been implemented in a production model GPS receiver.

1.5 Problem Statement

The complex cepstrum is a tool originally investigated for echo determination in seismic applications. By virtue of the fact that the cepstrum is a complex quantity, signal magnitude and phase information is preserved through the application of the complex cepstrum. The signal can subsequently be reconstructed through an inverse cepstrum process. Since multipath signals are "echoes" of the direct path signal (see Equation 6), the complex cepstrum will be investigated as a tool to remove the multipath interference in a composite signal. This thesis will pursue the complex cepstrum as a means to separate the multipath interference from the composite GPS signal. The multipath interference will then be removed by filtering (often called liftering) in the cepstral domain. Finally, the signal will be transformed back into the time domain for tracking by a standard GPS receiver. With the multipath interference thus removed, the induced tracking error will be removed, allowing accurate tracking of the direct path GPS signal.

1.6 Objectives

This thesis proposes a GPS receiver employing a complex cepstrum process for multipath removal, followed by a standard delay lock loop for code tracking. This new design modifies current GPS receivers by employing front end digital signal processing for the removal of multipath interference. The rest of the design is a standard GPS receiver.

The objectives of this thesis are:

1. Develop a complex cepstrum filtering technique for GPS multipath interference removal.
2. Modify a standard delay lock loop by adding the filtering to the loop input.
3. Characterize the multipath induced tracking error for the modified delay lock loop in a noiseless environment.
4. Characterize the multipath induced tracking error for the modified delay lock loop operating with typical GPS signal to noise ratios.
5. Compare and contrast the complex cepstrum receiver performance to that of a narrow correlator receiver.

1.7 Assumptions

For this thesis, the following assumptions are made:

1. The received signal consists of a direct path and a single reflected signal.
2. The receiver has completed the signal acquisition phase prior to multipath interference occurring.
3. The GPS signal has been down converted to an intermediate frequency.
4. The received signal is due to only one GPS satellite (i.e. one signal per channel).
5. No special antennas or spatial processing are used for multipath mitigation.
6. Doppler effects are negligible.

1.8 Approach

This thesis presents results of theoretical analysis and simulation of multipath mitigation using the complex cepstrum and cepstral domain filtering. Analyses and simulations are performed for a variety of multipath scenarios and in both noiseless and additive white Gaussian noise environments. Computer simulations are presented in Chapter IV. Simulations are written using MATLAB computational software from The Mathworks, Inc. of Natick, Massachusetts. Simulations are performed on a stand alone Pentium based PC and on the Sun workstations provided by the Air Force Institute of Technology (AFIT).

II. Cepstral Analysis

2.1 Introduction

In a 1963 paper by Bogert, Healy, and Tukey, entitled "The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudoautocovariance, Cross-Cepstrum, and Saphe Cracking" (1) it was observed that a signal composed of a fundamental and echoes of the fundamental could be decomposed by taking the logarithm of the power spectrum. Bogert et al. found that the logarithm of the power spectrum contained a periodic component due to the echoes in the composite signal. This component manifests itself as periodic peaks when the Fourier transform is taken. Because their technique uses the spectrum of the signal, Bogert et al. rearranged the letters of spectrum and in naming their new technique the "cepstrum". Since it's original development, many researchers have developed ways to modify the cepstrum for application to specific signal decomposition problems. To avoid confusion with these new techniques, what was originally called the "cepstrum" is now more commonly known as the "power cepstrum". Other forms of the cepstrum include the complex cepstrum, the phase cepstrum, and the log cepstrum. This thesis will use the power and complex cepstra. As a final note of introduction, complex cepstrum techniques fall into a class of nonlinear filtering techniques developed by Oppenheim, Schafer, and Stockham (7, 8) termed homomorphic deconvolution.

2.2 Definitions of the Power and Complex Cepstra

2.2.1 Power Cepstrum Definition. As previously mentioned, the power cepstrum was the first cepstral technique developed. This technique was used to estimate the delay and amplitude of echoes present in a composite signal. Since computer algorithms are typically employed to evaluate the power cepstrum, $x_{pc}[nT]$, it is usually written in terms of the Z-transform of a sampled signal as shown in Equation 8

$$\begin{aligned} x_{pc}[nT] &= \left(Z^{-1}\left(\log|X(z)|^2\right)\right)^2 \\ &= \left\{\frac{1}{2\pi j}\oint_c \log|X(z)|^2 z^{n-1}dz\right\}^2 \end{aligned} \tag{8}$$

where $X(z) \equiv \sum_{n=-\infty}^{\infty} x_n\, z^{-n}$ is the Z-transform of the discrete time signal $x[nT]$, n is the sample number, and T is sampling period. The logarithm can be computed relative to any base; however, the natural logarithm is most commonly used.

Two things should be noted concerning the power cepstrum. First, if $|z| = 1$ is in the region of convergence of $X(z)$, then the counter clockwise contour of integration is typically chosen to be the unit circle, so that the Z-transform becomes the discrete time Fourier transform (DTFT). Unless otherwise noted, the DTFT will be used throughout the rest of this thesis. Second, because the logarithm acts only on the magnitude of the Z-transform, the phase of the original signal is lost. Therefore, an inverse power cepstrum operation cannot be applied to recover the original signal. This weakness in the power cepstrum led to the development of the complex cepstrum which will now be presented.

2.2.2 Complex Cepstrum Definition. The complex cepstrum was developed out of the homomorphic system theory of Oppenheim, Schafer, and Stockham (7) and is very similar to the power cepstrum. The primary difference is that the complex cepstrum uses the complex logarithm of the Z-transform, maintaining the signal's phase information. The general definition of the complex logarithm and complex cepstrum, $\hat{x}[nT]$, are given in Equations 9 and 10, respectively.

$$\log c = \ln|c| + j\angle c \tag{9}$$

where c is a complex number, and $\angle c$ is the argument of the complex number, c.

$$\hat{x}[nT] = \frac{1}{2\pi j} \oint_c \log(X(z)) z^{n-1} dz \tag{10}$$

where, by definition, $\hat{x}[0] \equiv \log(\,x[0]\,)$. Finally, as a matter of notation, $\hat{X}(z)$ will be used to denote $\log X(z)$.

As mentioned earlier, the complex cepstrum is more versatile due to the fact that phase information is retained; however, carrying this phase information does create some difficulty in the calculation of the complex cepstrum. The difficulty arises from the fact that the complex logarithm is multi-valued in phase. If the phase of the logarithm is calculated modulo 2π (wherein $\angle c$ is the principal value of the argument of c), then phase discontinuities result; however, this is not allowed since $\log[X(z)]$ is the z-transform of $\hat{x}[nT]$ and thus must have a continuous phase. This difficulty is solved using any of several readily available phase unwrapping techniques. One such technique, presented by Childers, Skinner, and Kemerait (3) adds a correction term, $C(k)$, to the phase according to the following algorithm

$$C(k) = \begin{cases} 0 & \text{if} \quad k = 0 \\ C(k-1) - 2\pi & \text{if} \quad P(k) - P(k-1) > \pi \\ C(k-1) + 2\pi & \text{if} \quad P(k-1) - P(k) > \pi \\ C(k-1) & \quad \text{otherwise} \end{cases} \tag{11}$$

where $P(k)$ is the phase at point k.

2.3 Cepstral Processing of Multipath Signals

This section presents a typical use of the cepstrum, namely characterization of echoes (such as multipath) in a composite signal. First, the theory for using the power cepstrum is developed, followed by the theory of the complex cepstrum. Both of these derivations closely follow those found in (2, 3). Previous work has shown that it is often easier to detect multipath reflections in the power cepstrum, rather than the complex cepstrum (3). However, the complex cepstrum maintains the needed phase information contained in the signal. After calculating the complex cepstrum, a signal can be filtered in the cepstrum domain to remove the previously detected multipath reflection, then an inverse cepstral operation can be performed to recover the time domain direct path signal estimate. This process will be described in more detail later.

2.3.1 Power Cepstrum. To see how the power cepstrum can be used to decompose a signal comprised of a direct path and one multipath reflection, let $x[nT]$ be the signal of interest. This signal can be represented as follows

$$x[nT] = f[nT] * g[nT] \tag{12}$$

where $*$ represents the convolution operation, $f[nT]$ represents the direct path signal, and $g[nT]$ is defined as

$$g[nT] = \delta[nT] + a_0\delta[nT - n_0T] \tag{13}$$

Referring to Equation 8 for the definition of the power cepstrum, the steps in calculating the power cepstrum of $x[nT]$ are completed in the following manner

$$\begin{aligned} X(z) &= F(z)\,G(z) \qquad (14) \\ |X(z)| &= |F(z)| \cdot |G(z)| \\ |X(z)|^2 &= |F(z)|^2 \cdot |G(z)|^2 \\ &= |F(z)|^2 \,|\left(1 + a_0\, z^{-n_0}\right)|^2 \end{aligned}$$

Because $x[nT]$ is a causal, stable signal, the DTFT exists and is found by substitution of $z = e^{j\omega t}$:

$$\begin{aligned}
|X(e^{j\omega T})|^2 &= |F(e^{j\omega T})|^2 \, |\left(1 + a_0 \, e^{-j\omega n_0 T}\right)|^2 \\
\log|X(e^{j\omega T})|^2 &= \log|F(e^{j\omega T})|^2 + \log|\left(1 + a_0 \, e^{-j\omega n_0 T}\right)|^2 \\
&= \log|F(e^{j\omega T})|^2 + \log|1 + a_0 \cos(n_0\omega T) - j a_0 \sin(n_0\omega T))|^2 \\
&= \log|F(e^{j\omega T})|^2 + \log\{[1 + a_0 \cos(n_0\omega T)]^2 + [a_0 \sin(n_0\omega T)]^2\} \\
&= \log|F(e^{j\omega T})|^2 + \log[1 + 2a_0 \cos(n_0\omega T) + a_0^2 \cos^2(n_0\omega T) + a_0^2 \sin^2(n_0\omega T)] \\
&= \log|F(e^{j\omega T})|^2 + \log[1 + 2a_0 \cos(n_0\omega T) + a_0^2] \\
&= \log|F(e^{j\omega T})|^2 + \log[(1 + a_0^2)(1 + \frac{2a_0}{1 + a_0^2} \cos(n_0\omega T)] \\
&= \log|F(e^{j\omega T})|^2 + \log(1 + a_0^2) + \log[1 + \frac{2a_0}{1 + a_0^2} \cos(n_0\omega T)] \\
&= \log|F(e^{j\omega T})|^2 + \log(1 + a_0^2) + \log[1 + \frac{a_0}{1 + a_0^2}(e^{jn_0\omega T} + e^{-jn_0\omega T})] \qquad (15)
\end{aligned}$$

Next, use the power series expansion, $\log(1+x) = \sum_{k=1}^{\infty}(-1)^{k+1}\frac{x^k}{k}$, to expand the third term in Equation 15. Note that this expansion converges only if $|x| < 1$, which holds for this application. To prove this, let

$$x \equiv \frac{2\,a_0}{1 + a_0^2} \cos(n_0 \, \omega \, T)$$

Then,

$$0 \le (1 - a_0)^2 = 1 - 2\,a_0 + a_0^2$$

which implies

$$2\,a_0 \le 1 + a_0^2$$

or

$$\frac{2\,a_0}{1 + a_0^2} \le 1$$

Similarly, $(1 + a_0)^2 \geq 0$, leads to the conclusion

$$\frac{2\,a_0}{1 + a_0^2} \geq -1$$

Therefore,

$$-1 \leq \frac{2\,a_0}{1 + a_0^2} \leq 1 \tag{16}$$

Because $|\cos(n_0\,\omega\,T)| \leq 1$, if follows that

$$|x| = |\frac{2\,a_0}{1 + a_0^2}\cos(n_0\,\omega\,T)| \leq 1 \tag{17}$$

Therefore, the expansion is used for this problem and is given by

$$\log\left(1 + \frac{a_0}{1+a_0^2}(e^{jn_0\omega T} + e^{-jn_0\omega T})\right) = \sum_{m=1}^{\infty}\frac{(-1)^{m+1}}{m}\left[\frac{a_0}{1+a_0^2}(e^{jn_0\omega T} + e^{-jn_0\omega T})\right]^m \tag{18}$$

Combining Equations 15 and 18 gives

$$\begin{aligned}
\log|X(e^{j\omega T})|^2 &= \log|F(e^{j\omega T})|^2 + \log(1+a_0^2) + \sum_{m=1}^{\infty}\frac{(-1)^{m+1}}{m}[\frac{a_0}{1+a_0^2}(e^{jn_0\omega T} + e^{-jn_0\omega T})]^m \quad (19)\\
&= \log|F(e^{j\omega T})|^2 + \log(1+a_0^2) + \frac{a_0}{1+a_0^2}(e^{jn_0\omega T} + e^{-jn_0\omega T})\\
&\quad -\frac{1}{2}\left(\frac{a_0}{1+a_0^2}\right)^2(e^{jn_0\omega T} + e^{-jn_0\omega T})^2 + \frac{1}{3}\left(\frac{a_0}{1+a_0^2}\right)^3(e^{jn_0\omega T} + e^{-jn_0\omega T})^3 - \ldots\\
&= \log|F(e^{j\omega T})|^2 + \log(1+a_0^2) + \frac{a_0}{1+a_0^2}(e^{jn_0\omega T} + e^{-jn_0\omega T})\\
&\quad -\frac{1}{2}\left(\frac{a_0}{1+a_0^2}\right)^2(e^{j2n_0\omega T} + e^{-j2n_0\omega T} + 2)\\
&\quad +\frac{1}{3}\left(\frac{a_0}{1+a_0^2}\right)^3(e^{j3n_0\omega T} + e^{-j3n_0\omega T} + 3e^{jn_0\omega T} + 3e^{-jn_0\omega T}) - \ldots
\end{aligned}$$

Note that each term with an even exponent in the expansion will contain a constant, with decreasing magnitude as the exponent increases. Also, note that the term $\log(1 + a_0^2)$ is a constant. Lumping these constants together into another constant called K, and taking the inverse Fourier transform, gives the power cepstrum of this signal.

$$
\begin{aligned}
x_{pc}[nT] &= F^{-1}\{\log|F(e^{j\omega T})|^2\} + F^{-1}\{K\} + F^{-1}\{\ \frac{a_0}{1+a_0^2}(e^{jn_0\omega T} + e^{-jn_0\omega T}) \\
&\quad -\frac{1}{2}\left(\frac{a_0}{1+a_0^2}\right)^2 (e^{j2n_0\omega T} + e^{-j2n_0\omega T} + 2) \\
&\quad +\frac{1}{3}\left(\frac{a_0}{1+a_0^2}\right)^3 (e^{j3n_0\omega T} + e^{-j3n_0\omega T} + 3e^{jn_0\omega T} + 3e^{-jn_0\omega T}) - \ldots\ \} \\
&= F^{-1}\{\log|F(e^{j\omega T})|^2\} + K\delta(t) + \frac{a_0}{1+a_0^2}\left[\delta(t+n_0T) + \delta(t-n_0T)\right] \\
&\quad -\frac{1}{2}(\frac{a_0}{1+a_0^2})^2\left[\delta(t+2n_0T) - \delta(t-2n_0T)\right] \\
&\quad +\frac{1}{3}(\frac{a_0}{1+a_0^2})^3\left[\delta(t+3n_0T) + \delta(t-3n_0T)\right] - \ldots \qquad (20)
\end{aligned}
$$

Examination of Equation 20 shows why the power cepstrum is useful for determining the delay of a multipath signal. The power cepstrum of the composite signal exhibits periodic peaks which are delayed at integer multiples of the multipath delay; thus, the power cepstrum can be used to detect a multipath component in a composite signal. The difficulty occurs when one attempts to characterize multiple multipath reflections. This difficulty is demonstrated in the derivation that follows, which parallels a derivation in (3).

The two echo case is derived following a procedure similar to that for a single echo. In this derivation, arbitrarily many reflections could have been used. However, the two reflection scenario is useful for highlighting the key points without adding unnecessary complexity.

First, refer to Equation 12 as the starting point. Then, rewrite Equation 13 to account for two reflections, which yields Equation 21

$$g[nT] = \delta[nT] + a_0\delta[nT - n_0T] + a_1\delta[nT - n_1T] \qquad (21)$$

where a_i and n_iT are the amplitude and delay, respectively, of the i^{th} reflection.

Next, follow the same steps as in the previous derivation.

$$
\begin{align}
X(z) &= F(z)\cdot G(z) \tag{22}\\
|X(z)| &= |F(z)|\cdot|G(z)| \notag\\
|X(z)|^2 &= |F(z)|^2\cdot|G(z)|^2 \notag\\
&= |F(z)|^2\,\big|\left(1+a_0\,z^{-n_0}+a_1\,z^{-n_1}\right)\big|^2 \tag{23}
\end{align}
$$

Evaluating on the unit circle, $z = e^{j\omega t}$ and taking the logarithm gives

$$
\begin{aligned}
|X(e^{j\omega T})|^2 &= |F(e^{j\omega T})|^2\,\big|\left(1+a_0\,e^{-jn_0\omega T}+a_1\,e^{-jn_1\omega T}\right)\big|^2\\
\log|X(e^{j\omega T})|^2 &= \log|F(e^{j\omega T})|^2+\log\big|\left(1+a_0\,e^{-jn_0\omega T}+a_1\,e^{-jn_1\omega T}\right)\big|^2\\
&= \log|F(e^{j\omega T})|^2\\
&\quad +\log|1+a_0\cos(n_0\omega T)+a_1\cos(n_1\omega T)-j[a_0\sin(n_0\omega T)+a_1\sin(a_1\omega T)]|^2\\
&= \log|F(e^{j\omega T})|^2\\
&\quad +\log\{[1+a_0\cos(n_0\omega T)+a_1\cos(n_1\omega T)]^2+[a_0\sin(n_0\omega T)+a_1\sin(n_1\omega T)]^2\}\\
&= \log|F(e^{j\omega T})|^2+\log[1+2a_0\cos(n_0\omega T)+2a_1\cos(n_1\omega T)+a_0^2\cos^2(n_0\omega T)\\
&\quad +2a_0a_1\cos(n_0\omega T)\cos(n_1\omega T)+a_1^2\cos^2(n_1\omega T)+a_0^2\sin^2(n_0\omega T)\\
&\quad +2a_0a_1\sin(n_0\omega T)\sin(n_1\omega T)+a_1^2\sin(n_1\omega T)]\\
&= \log|F(e^{j\omega T})|^2+\log[1+a_0^2+a_1^2+2a_0\cos(n_0\omega T)+2a_1\cos(n_1\omega T)\\
&\quad +2a_0a_1\{\cos[(n_1-n_0)\omega T]+\cos[(n_1+n_0)\omega T]\\
&\quad +\cos[(n_1-n_0)\omega T]-\cos[(n_1+n_0)\omega T]\}]\\
&= \log|F(e^{j\omega T})|^2\\
&\quad +\log\{1+a_0^2+a_1^2+2a_0\cos(n_0\omega T)+2a_1\cos(n_1\omega T)+2a_0a_1\cos[(n_1-n_0)\omega T]\}
\end{aligned}
$$

$$
\begin{aligned}
&= \log|F(e^{j\omega T})|^2 + \log\{\,[1+a_0^2+a_1^2][1+\frac{2a_0}{1+a_0^2+a_1^2}\cos(n_0\omega T)\\
&\quad +\frac{2a_1}{1+a_0^2+a_1^2}\cos(n_1\omega T)+\frac{2a_0a_1}{1+a_0^2+a_1^2}\cos\{(n_1-n_0)\omega T\}]\,\}\\
&= \log|F(e^{j\omega T})|^2+\log(1+a_0^2+a_1^2)\\
&\quad +\log[\,1+\frac{2a_0}{1+a_0^2+a_1^2}\cos(n_0\omega T)+\frac{2a_1}{1+a_0^2+a_1^2}\cos(n_1\omega T)\\
&\quad +\frac{2a_0a_1}{1+a_0^2+a_1^2}\cos\{(n_1-n_0)\omega T\}\,]\\
&= \log|F(e^{j\omega T})|^2+\log(1+a_0^2+a_1^2)+\log\{\,1+\frac{1}{1+a_0^2+a_1^2}[\,a_0(e^{jn_0\omega T}+e^{-jn_0\omega T})\\
&\quad +a_1(e^{jn_1\omega T}+e^{-jn_1\omega T})+a_0a_1(e^{j(n_1-n_0)\omega T}+e^{-j(n_1-n_0)\omega T})\,]\,\} \qquad (24)
\end{aligned}
$$

Next, the final term of Equation 24 can be expanded in a similar manner as for the single reflection case. Incorporating the expansion into Equation 24 gives

$$
\begin{aligned}
F^{-1}\{\log|X(e^{j\omega T})|^2\} &= F^{-1}\{\log|F(e^{j\omega T})|^2\}+F^{-1}\{\log(1+a_0^2+a_1^2)\}\\
&\quad +F^{-1}\{\sum_{k=1}^{\infty}\frac{1}{k}\,(-1)^{k+1}\left\{\frac{1}{1+a_0^2+a_1^2}\right\}^k\cdot\\
&\quad \Big[a_0(e^{jn_0\omega T}+e^{-jn_0\omega T})+a_1(e^{jn_1\omega T}+e^{-jn_1\omega T})\\
&\quad +a_0a_1(e^{j(n_1-n_0)\omega T}+e^{-j(n_1-n_0)\omega T})\Big]^k\\
&= F^{-1}\{\log|F(e^{j\omega T})|^2\}+F^{-1}\{\log(1+a_0^2+a_1^2)\}\\
&\quad +F^{-1}\{\frac{1}{1+a_0^2+a_1^2}[a_0(e^{jn_0\omega T}+e^{-jn_0\omega T})+a_1(e^{jn_1\omega T}+e^{-jn_1\omega T})\\
&\quad +a_0a_1(e^{j(n_1-n_0)\omega T}+e^{-j(n_1-n_0)\omega T})]\\
&\quad -\frac{1}{2}(\frac{1}{1+a_0^2+a_1^2})^2[a_0(e^{jn_0\omega T}+e^{-jn_0\omega T})+a_1(e^{jn_1\omega T}+e^{-jn_1\omega T})\\
&\quad +a_0a_1(e^{j(n_1-n_0)\omega T}+e^{-j(n_1-n_0)\omega T})]^2\\
&\quad +\frac{1}{3}(\frac{1}{1+a_0^2+a_1^2})^3[a_0(e^{jn_0\omega T}+e^{-jn_0\omega T})+a_1(e^{jn_1\omega T}+e^{-jn_1\omega T})\\
&\quad +a_0a_1(e^{j(n_1-n_0)\omega T}+e^{-j(n_1-n_0)\omega T})]^3-\ldots\}\\
&= F^{-1}\{\log|F(e^{j\omega T})|^2\}+F^{-1}\{\,\log(1+a_0^2+a_1^2)\}\\
&\quad +F^{-1}\{\frac{1}{1+a_0^2+a_1^2}[a_0(e^{jn_0\omega T}+e^{-jn_0\omega T})+a_1(e^{jn_1\omega T}+e^{-jn_1\omega T})
\end{aligned}
$$

$$
\begin{aligned}
&+a_0a_1(e^{j(n_1-n_0)\omega T}+e^{-j(n_1-n_0)\omega T})]\\
&-\frac{1}{2}(\frac{1}{1+a_0^2+a_1^2})^2[a_0^2(e^{j2n_0\omega T}+e^{-j2n_0\omega T}+2)\\
&+a_1^2(e^{j2n_1\omega T}+e^{-j2n_1\omega T}+2)\\
&+a_0^2a_1^2(e^{j2(n_1-n_0)\omega T}+e^{-j2(n_1-n_0)\omega T}+2)\\
&+a_0a_1(e^{j(n_1+n_0)\omega T}+e^{-j(n_1+n_0)\omega T}+e^{j(n_1-n_0)\omega T}+e^{-j(n_1-n_0)\omega T})\\
&+a_0^2a_1(e^{jn_1\omega T}+e^{-jn_1\omega T}+e^{j(n_1-2n_0)\omega T}+e^{-j(n_1-2n_0)\omega T})\\
&+a_0a_1^2(e^{jn_0\omega T}+e^{-jn_0\omega T}+e^{j(2n_1-n_0)\omega T}+e^{-j(2n_1-n_0)\omega T})]\\
&+\frac{1}{3}(\frac{1}{1+a_0^2+a_1^2})^3[a_0^3(e^{j3n_0\omega T}+e^{-j3n_0\omega T}+3e^{jn_0\omega T}+3e^{-jn_0\omega T})\\
&+a_1^3(e^{j3n_1\omega T}+e^{-j3n_1\omega T}+3e^{jn_1\omega T}+3e^{-jn_1\omega T})\\
&+a_0^3a_1^3(e^{j3(n_1-n_0)\omega T}+e^{-j3(n_1-n_0)\omega T}+3e^{jn_0\omega T}+3e^{-jn_0\omega T})\\
&+3a_0^2a_1(2e^{jn_1\omega T}+2e^{-jn_1\omega T}+e^{j(2n_0+n_1)\omega T}+e^{-j(2n_0+n_1)\omega T}\\
&+e^{j(2n_0-n_1)\omega T}+e^{-j(2n_0-n_1)\omega T})+3a_0a_1^2(2e^{jn_0\omega T}+2e^{-jn_0\omega T}\\
&+e^{j(2n_1+n_0)\omega T}+e^{-j(2n_1+n_0)\omega T}+e^{j(2n_1-n_0)\omega T}+e^{-j(2n_1-n_0)\omega T})\\
&+3a_0a_1^3(2e^{j(n_1-n_0)\omega T}+2e^{-j(n_1-n_0)\omega T}+e^{j(3n_1-n_0)\omega T}+e^{-j(3n_1-n_0)\omega T}\\
&+e^{j(n_1+n_0)\omega T}+e^{-j(n_1+n_0)\omega T})+3a_0^2a_1^3(2e^{jn_1\omega T}+2e^{-jn_1\omega T}\\
&+e^{j(3n_1-2n_0)\omega T}+e^{-j(3n_1-2n_0)\omega T}+e^{j(n_1-2n_0)\omega T}+e^{-j(n_1-2n_0)\omega T})\\
&+a_0^3a_1(2e^{j(n_1-n_0)\omega T}+2e^{-j(n_1-n_0)\omega T}+e^{j(n_1+n_0)\omega T}+e^{-j(n_1+n_0)\omega T}\\
&+e^{j(n_1-3n_0)\omega T}+e^{-j(n_1-3n_0)\omega T})+a_0^3a_1^2(2e^{jn_0\omega T}+2e^{-jn_0\omega T}+e^{j(2n_1-3n_0)\omega T}\\
&+e^{-j(2n_1-n_0)\omega T}+e^{j(2n_1-n_0)\omega T}+e^{-j(2n_1-n_0)\omega T})+4a_0^2a_1^2(e^{j2n_0\omega T}+e^{-j2n_0\omega T}\\
&+e^{j2n_1\omega T}+e^{-j2n_1\omega T}+e^{j2(n_1-n_0)\omega T}+e^{-j2(n_1-n_0)\omega T}+2)]-\ldots\}
\end{aligned}
\tag{25}
$$

Next, the inverse Fourier transforms are evaluated to yield the power cepstrum. After collecting like terms, the equation above becomes the following.

$$
\begin{aligned}
x_{pc}[nT] \quad = \quad & F^{-1}\{\log|F(e^{j\omega T})|^2\} \\
& + \left[\log(1+a_0^2+a_1^2) + \frac{a_0^4(1+a_1^2)+a_1^4(1+a_0^2)+3a_0^2a_1^2+a_0^2+a_1^2}{(1+a_0^2+a_1^2)^3}\right]\delta(t) \\
& + \frac{6a_0^5+18a_0^3+6a_0+a_0^3(6a_1^3+13a_1^2)+a_0(3a_1^4+21a_1^2)}{6(1+a_0^2+a_1^2)^3}\left[\delta(t-n_0T)+\delta(t+n_0T)\right] \\
& + \left[\frac{6a_1^5+12a_1^3+6a_0^3+6a_1-a_0(3a_1^2-6a_1)+a_0^3(6a_1^3+4a_1^2)}{6(1+a_0^2+a_1^2)^3}\right. \\
& + \left.\frac{-a_0^2(3a_1^4-12a_1^3+3a_1^2-12a_1)+12a_0a_1}{6(1+a_0^2+a_1^2)^3}\right]\left[\delta(t-n_1T)+\delta(t+n_1T)\right] \\
& + \frac{5a_0^2a_1^2-3a_0^4-3a_0^2}{6(1+a_0^2+a_1^2)^3)}\left[\delta(t-2n_0T)+\delta(t+2n_0T)\right] \\
& + \frac{5a_0^2a_1^2-3a_1^4-3a_1^2}{6(1+a_0^2+a_1^2)^3)}\left[\delta(t-2n_1T)+\delta(t+2n_1T)\right] \\
& + \frac{a_0^3}{3(1+a_0^2+a_1^2)^3}\left[\delta(t-3n_0T)+\delta(t+3n_0T)\right] \\
& + \frac{a_1^3}{3(1+a_0^2+a_1^2)^3}\left[\delta(t-3n_1T)+\delta(t+3n_1T)\right] \\
& + \frac{6a_0^5a_1+a_0(12a_1^3+17a_1)+a_0(6a_1^5+21a_1^3+3a_1)}{6(1+a_0^2+a_1^2)^3}\left[\delta(t-(n_1-n_0)T)+\delta(t+(n_1-n_0)T)\right] \\
& + \frac{5a_0^2a_1^2-3a_0^4a_1^2-3a_0^2a_1^4}{6(1+a_0^2+a_1^2)^3}\left[\delta(t-2(n_1-n_0)T)+\delta(t+2(n_1-n_0)T)\right] \\
& + \frac{a_0^3a_1^3}{3(1+a_0^2+a_1^2)^3}\left[\delta(t-3(n_1-n_0)T)+\delta(t+3(n_1-n_0)T)\right] \\
& + \frac{3a_0(a_1^3-a_1)-a_0^3a_1}{6(1+a_0^2+a_1^2)^3}\left[\delta(t-(n_1+n_0)T)+\delta(t+(n_1+n_0)T)\right] \\
& + \frac{2a_0a_1^3-3a_0^4a_1-3a_0^2(a_1^3+a_1)}{6(1+a_0^2+a_1^2)^3}\left[\delta(t-(n_1-2n_0)T)+\delta(t+(n_1-2n_0)T)\right] \\
& + \frac{3a_0(a_1^2-a_1^3)-a_0^3a_1^2}{6(1+a_0^2+a_1^2)^3}\left[\delta(t-(2n_1-n_0)T)+\delta(t+(2n_1-n_0)T)\right] \\
& + \frac{a_0^2a_1}{(1+a_0^2+a_1^2)^3}\left[\delta(t-(2n_0+n_1)T)+\delta(t+(2n_0+n_1)T)\right] \\
& + \frac{a_0^2a_1}{(1+a_0^2+a_1^2)^3}\left[\delta(t-(2n_0-n_1)T)+\delta(t-(2n_0+n_1)T)\right] \\
& + \frac{a_0a_1^2}{(1+a_0^2+a_1^2)^3}\left[\delta(t-(2n_1+n_0)T)+\delta(t+(2n_1+n_0)T)\right] \\
& + \frac{a_0a_1^3}{(1+a_0^2+a_1^2)^3}\left[\delta(t-(3n_1-n_1)T)+\delta(t+(3n_1-n_0)T)\right]
\end{aligned}
\tag{26}
$$

$$+\frac{a_0^2 a_1^3}{(1+a_0^2+a_1^2)^3}\left[\delta(t-(3n_1-2n_0)T)+\delta(t+(3n_1-2n_0)T)\right]$$
$$+\frac{a_0^3 a_1}{3(1+a_0^2+a_1^2)^3}\left[\delta(t-(n_1-3n_0)T)+\delta(t+(n_1-3n_0)T)\right]$$
$$+\frac{a_0^3 a_1^2}{3(1+a_0^2+a_1^2)^3}\left[\delta(t-(2n_1-3n_0)T)+\delta(t+(2n_1-3n_0)T)\right] \; + \; \ldots$$

From Equation 26 it is seen that the power cepstrum will contain peaks at the delays of each reflection, just as it did in the single reflection case. However, it is also seen that the reflections will interact with each other causing peaks at sums and differences of the delays. It is these peaks that cause the difficulty in determining the true multipath delays. If a sufficiently small number of multipath reflections exist, the true delays can be distinguished from these interference delays using an exhaustive search. This method would involve comparing the delay associated with every peak to the sums and differences of the other delays, and determining which delays are true multipath delays. Using a computer algorithm, this procedure could probably be implemented for cases where only a few reflections exist. However, this method would become very time consuming if many reflections exist. Therefore, rather than dealing with this difficulty, this thesis will focus on using the complex cepstrum to remove all reflections without necessarily estimating the parameters associated with the reflections.

2.3.2 Power Cepstrum Example. To see how the power cepstrum can be used to determine the delay in a single multipath reflection case, consider the following simple example. Let $x[nT]$ be the direct path signal

$$x[nT] = e^{-nT} \tag{27}$$

where T is the sampling period, n is the sample number, and the signal is given an initial amplitude of 1 so that everything is normalized to the direct path.

Next, let a single multipath reflection occur three time units later than the direct path, with an amplitude of one half with respect to the direct path. Then, from Equation 6 the composite signal is given by

$$c[nT] = e^{-nT} + 0.5e^{-(nT-3)} \tag{28}$$

where $c[nT]$ is the composite signal. A plot of these signals is shown in Figure 3.

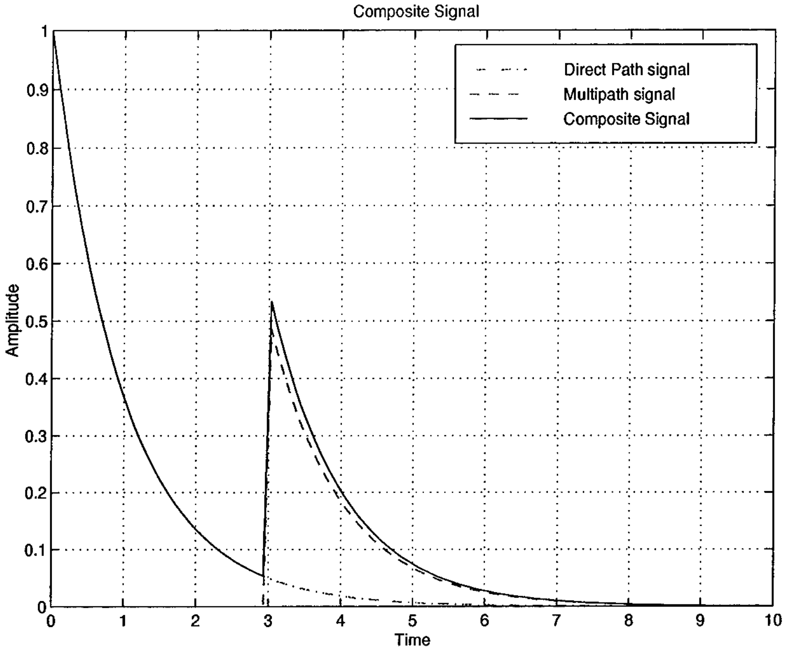

Figure 3 Signals for Power Cepstrum Example

Following the procedures previously developed, the power cepstrum of the composite signal was calculated and is shown in Figure 4. It is seen in Figure 4 that the power cepstrum contains peaks at ±3 time units, as predicted by Equation 20.

2.3.3 Complex Cepstrum. The development of the complex cepstrum in a multipath environment is very similar to that of the power cepstrum. The difference lies in the fact that the complex logarithm is used to retain phase information. As mentioned earlier, this allows one to filter the signal in the cepstral domain, then perform an inverse operation to return to the time domain. This process is shown in Figure 5. The single reflection complex cepstrum derivation follows. As before, this derivation parallels those in (2, 3).

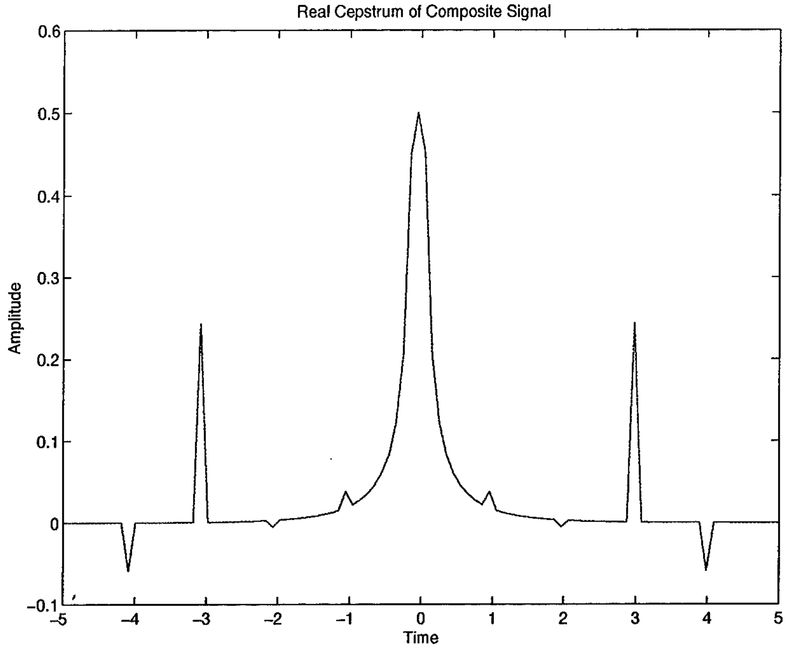

Figure 4 Power Cepstrum for Example Signal

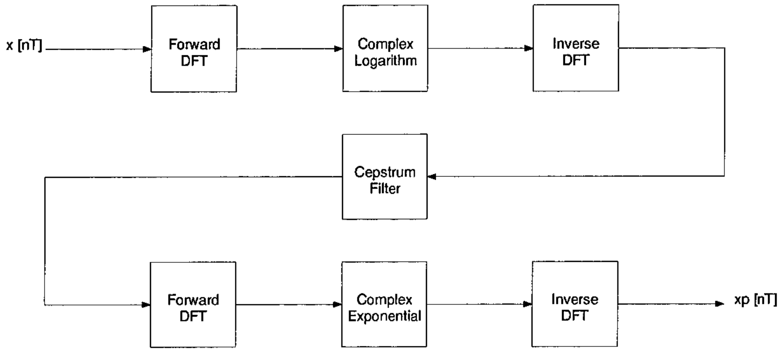

Figure 5 Complex Cepstrum Filtering Process

For the complex cepstrum derivation, begin with a signal which is the convolution of two other signals.

$$x[nT] = f[nT] * g[nT] \tag{29}$$

where $*$ represents the convolution operation, $f[nT]$ represents the direct path signal, and, for a single reflection, $g[nT]$ is defined as

$$g[nT] = \delta[nT] + a_0\delta[nT - n_0T] \tag{30}$$

Referring to Equation 10, the steps in computing the complex cepstrum are completed as follows.

$$\begin{aligned} X(z) &= F(z)G(z) \\ &= F(z)(1+a_0 z^{-n_0}) \end{aligned} \tag{31}$$

Evaluating the Z-transform on the unit circle and taking the logarithm yields

$$X(e^{j\omega T}) = F(e^{j\omega T})(1+a_0 e^{-j\omega n_0 T}) \tag{32}$$

$$\log[X(e^{j\omega T})] = \log[F(e^{j\omega T})] + \log[(1+a_0 e^{-j\omega n_0 T})] \tag{33}$$

Again, a power series expansion is used for the final term above, provided that $|a_0\, e^{-j\omega n_0 T}| = |a_0| < 1$, which requires that reflections be attenuated relative to the direct path signal. This generally is true due to the conservation of energy, but can be violated when many reflections combine in phase to form a composite reflection of greater magnitude.

$$\log[(1+a_0 e^{-j\omega T})] = \sum_{k=1}^{\infty}(-1)^{k+1}\frac{(a_0 e^{-j\omega n_0 T})^k}{k} \tag{34}$$

Using the expansion yields

$$\log[X(e^{j\omega T})] = \log[F(e^{j\omega T})] + \sum_{k=1}^{\infty}(-1)^{k+1}\frac{(a_0 e^{-j\omega n_0 T})^k}{k} \tag{35}$$

Finally, the inverse Fourier transform is taken to give the complex cepstrum.

$$\begin{aligned} F^{-1}\{\log[X(e^{j\omega T})]\} &= F^{-1}\{\log[F(e^{j\omega T})]\} \\ &\quad +F^{-1}\left\{a_0 e^{-j\omega n_0 T} - \frac{a_0^2}{2}e^{-j2\omega n_0 T} + \frac{a_0^3}{3}e^{-j3\omega n_0 T} - \ldots\right\} \qquad (36) \\ &= F^{-1}\{\log[F(e^{j\omega T})]\} + a_0\delta(t-n_0 T) - \frac{a_0^2}{2}\delta(t-2n_0 T) + \frac{a_0^3}{3}\delta(t-3n_0 T) - \ldots \end{aligned}$$

Just as in the power cepstrum case, it is seen that the complex cepstrum will contain peaks at integer multiples of the multipath delay. However, unlike with the power cepstrum, these peaks can be removed from the complex cepstrum by filtering. Following this filtering operation with an inverse complex cepstrum will then yield a multipath free signal in the time domain.

The same procedure as above can be followed for the multiple echo scenario (3). Again, as in the power cepstrum case, a two echo case will be presented for illustration purposes. However, the analysis could be extended to the case of arbitrarily many echoes.

Again, start with a signal which is the convolution of two signals, as given in Equation 29. However, in this case define $g[nT]$ as

$$g[nT] = \delta[nT] + a_0\delta[nT - n_0T] + a_1\delta[nT - n_1T] \tag{37}$$

where a_i and n_iT are the amplitude and delay, respectively, of the i^{th} multipath signal.

Then, the derivation follows as before.

$$\begin{aligned} X(z) &= F(z)G(z) \qquad (38) \\ &= F(z)(1 + a_0z^{-n_0} + a_1z^{-n_1}) \end{aligned}$$

Evaluating the Z-transforms on the unit circle and taking the logarithm yields

$$\begin{aligned} X(e^{j\omega T}) &= F(e^{j\omega T})(1 + a_0e^{-j\omega n_0T} + a_1e^{-j\omega n_1T}) \qquad (39) \\ \log[X(e^{j\omega T})] &= \log[F(e^{j\omega T})] + \log[(1 + a_0e^{-j\omega n_0T} + a_1e^{-j\omega n_1T})] \end{aligned}$$

Using the power series expansion for the last term, when $|\, a_0\, e^{-j\omega n_0T} + a_1\, e^{-j\omega n_1T}\,| < 1$, gives the following

$$\log[X(e^{j\omega T})] = \log[F(e^{j\omega T})] + \sum_{k=1}^{\infty}(-1)^{k+1}\frac{(a_0e^{-j\omega n_0T} + a_1e^{-j\omega n_1T})^k}{k} \tag{40}$$

Finally, taking the inverse Fourier transform yields the complex cepstrum.

$$\begin{aligned} F^{-1}\{\log[X(e^{j\omega T})]\} &= F^{-1}\{\log[F(e^{j\omega T})]\} \qquad (41) \\ &= +F^{-1}\{\; a_0 e^{-j\omega n_0 T} + a_1 e^{-j\omega n_1 T} - \frac{(a_0 e^{-j\omega n_0 T} + a_1 e^{-j\omega n_1 T})^2}{2} \\ &\quad + \frac{(a_0 e^{-j\omega n_0 T} + a_1 e^{-j\omega n_1 T})^3}{3} - \ldots \} \\ &= F^{-1}\{\log[F(e^{j\omega T})]\} + F^{-1}\{\; a_0 e^{-j\omega n_0 T} + a_1 e^{-j\omega n_1 T} - \frac{1}{2} a_0^2 e^{-j\omega n_0 T} \\ &\quad - \frac{1}{2} a_1^2 e^{-j\omega n_1 T} - \frac{1}{2} a_0 a_1 e^{-j\omega(n_0+n_1)T} + \frac{1}{3} a_0^3 e^{-j3\omega n_0 T} + \frac{1}{3} a_1^3 e^{-j3\omega n_1 T} \\ &\quad + a_0^2 a_1 e^{-j\omega(2n_0+n_1)T} + a_0 a_1^2 e^{-j\omega(n_0+2n_1)T} \ldots \} \\ &= F^{-1}\{\log[F(e^{j\omega T})]\} + a_0 \delta(t - n_0 T) + a_1 \delta(t - n_1 T) \\ &\quad - \frac{1}{2} a_0^2 \delta(t - 2n_0 T) - \frac{1}{2} a_1^2 \delta(t - 2n_1 T) - a_0^2 a_1 \delta(t - (n_0 + n_1)T) \\ &\quad + \frac{1}{3} a_0^3 \delta(t - 3n_0 T) + \frac{1}{3} a_1^3 \delta(t - 3n_1 T) + a_0^2 a_1 \delta(t - (2n_0 + n_1)T) \\ &\quad + a_0 a_1^2 \delta(t - (n_0 + 2n_1)T) - \ldots \end{aligned}$$

As in the single reflection case, it is seen from Equation 41 that the complex cepstrum will contain peaks with amplitudes and delays proportional to those of the multipath signals. Also, like the power cepstrum, it is seen that the multiple reflections will interact causing difficulty in using the complex cepstrum to characterize the multipath signals. However, unlike in the power cepstrum case, the complex cepstrum can be filtered in the cepstral domain, then inverted back to the time domain. Using this technique, the multipath signals are removed and the direct path signal recovered without needing to fully characterize the multipath signals.

2.4 Filtering in the Cepstral Domain

Three methods for filtering a cepstral domain signal are long pass, short pass, and comb filtering. Each of these methods has advantages and disadvantages depending upon the goals of the designer. The long pass filter is the cepstral equivalent to a frequency domain high pass filter.

The long pass filter is designed to set all points in the cepstrum prior to the first echo peak equal to zero. Thus using a long pass filter preserves the multipath signals while filtering out the direct path signal. Similarly, the short pass filter is the cepstral equivalent of a low pass filter. The short pass filter replaces everything from the first echo peak forward (in time) with a zero. Thus, the short pass filter removes all multipath signals while passing the direct path signal. The disadvantage of both the long pass and short pass filters is that, by zeroing out large portions of the complex cepstrum, these filters may cause significant degradation to the signal(s) of interest. An alternative to these methods is the comb filter. This filter is the cepstral equivalent to a notch filter. The comb filter replaces multipath peaks in the complex cepstrum with the average of the two (or more) points immediately adjacent to the peaks. In this manner, the comb filter removes the delta functions in the complex cepstrum which are caused by the multipath interference. Thus, when the filtered signal is converted back to the time domain, the direct path signal is recovered. The difficulty with this technique lies in detecting the peaks in the complex cepstrum. As previously mentioned, researchers (3) have found that it is often easier to detect multipath peaks by using the power cepstrum rather than the complex cepstrum. A parallel process is employed to detect and mitigate multipath effects; the power cepstrum is used to detect multipath delays and the complex cepstrum is used to filter out the multipath reflection. A block diagram of this process is shown in Figure 6. A simple example using the complex cepstrum to filter out multipath follows.

2.4.1 Multipath Mitigation Example. As an example of the theory that has been presented consider again the decaying exponential signal, $x[nT]$

$$x[nT] = e^{-nT} \tag{42}$$

where T is the sampling period, n is the sample number, and the signal was given an initial amplitude of 1 so that everything is normalized to this signal.

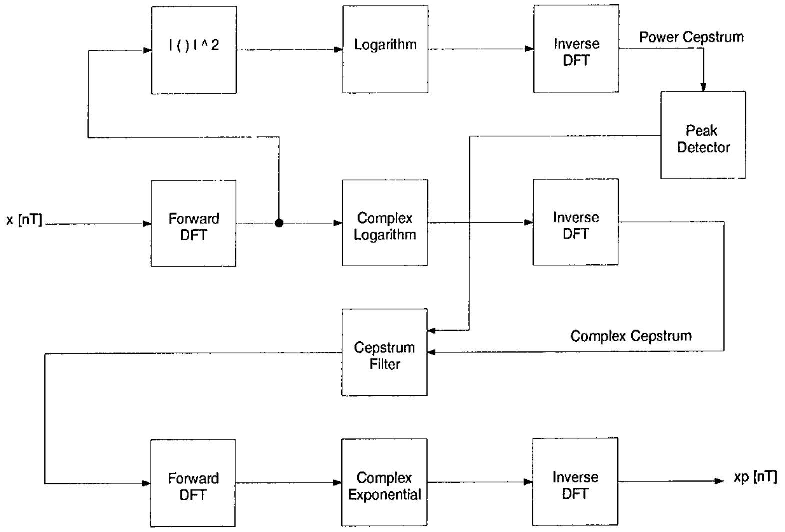

Figure 6 Power and Complex Cepstrum Multipath Detection and Mitigation Procedure

If $x[nT]$ is the direct path signal, and a multipath reflection occurs three time units later with an amplitude of one half with respect to the direct path, then the composite signal is represented as follows.

$$\begin{aligned} c[nT] &= x[nT] + 0.5x[nT-3] \\ c[nT] &= e^{-nT} + 0.5e^{-(nT-3)} \end{aligned} \tag{43}$$

where $c[nT]$ is the composite signal. A plot of these signals is shown in Figure 7.

Recall that the procedure for removing the multipath signal is shown in Figure 5. The first step is to calculate the complex cepstrum of the composite signal as previously presented. A plot of the complex cepstrum for this example is shown in Figure 8. Note that the complex cepstrum contains peaks at 3, 6, and 9 time units as predicted by the theory previously presented. The next step in removing the multipath signal is to comb filter the complex cepstrum at time units 3, 6, and 9. This is accomplished by replacing complex cepstrum values at those times with the averages of the two points on either side of the peaks. The complex cepstrum of the multipath signal and the comb filtered complex cepstrum are both shown in Figure 8. Finally, the comb filtered complex

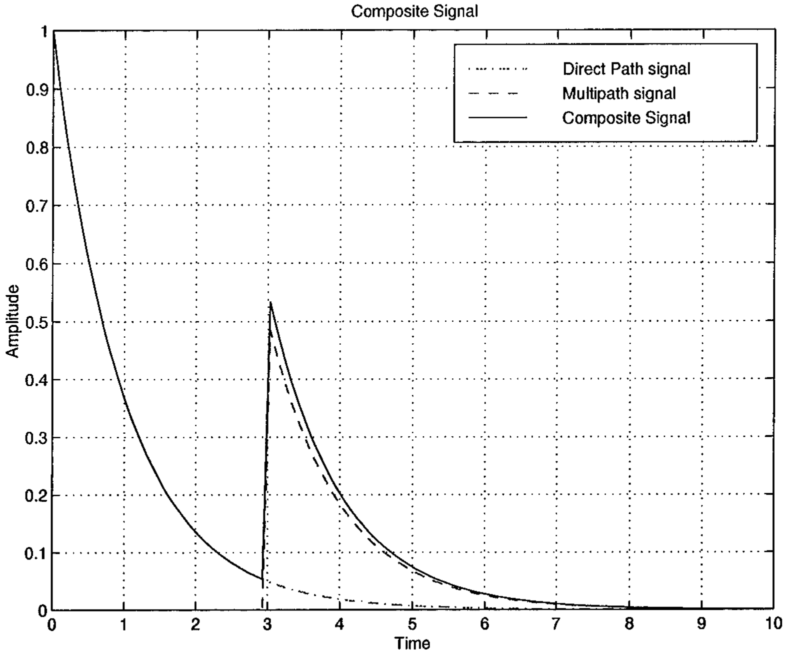

Figure 7 Signals for Complex Cepstrum Example

cepstrum is transformed back into the time domain using an inverse complex cepstrum operation to recover the direct path signal. The recovered direct path signal for this example is shown in Figure 9 along with a plot of the original direct path signal. The mean squared error between the recovered direct path signal and the actual direct path signal is 1.16×10^{-6}, demonstrating that this method is effective for multipath mitigation for this signal.

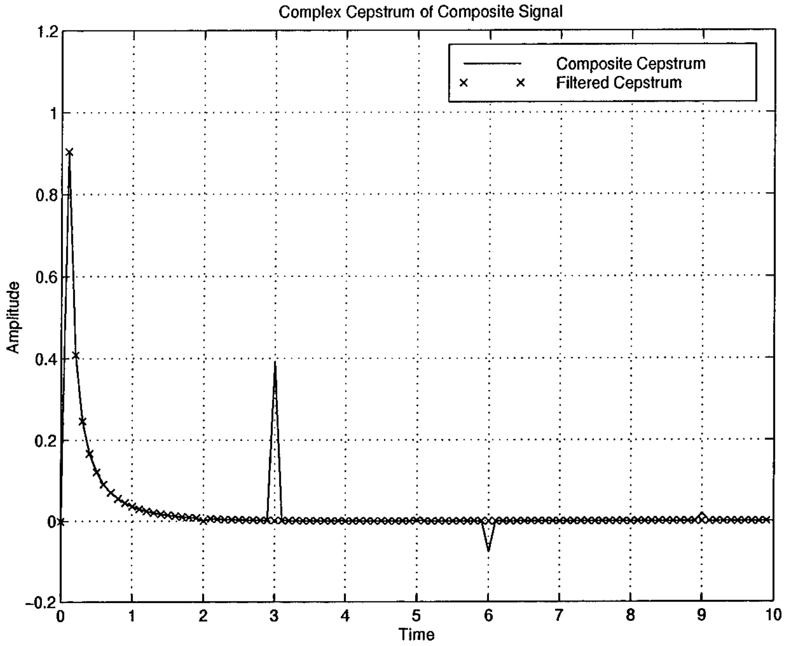

Figure 8 Complex Cepstrum for Example Signal

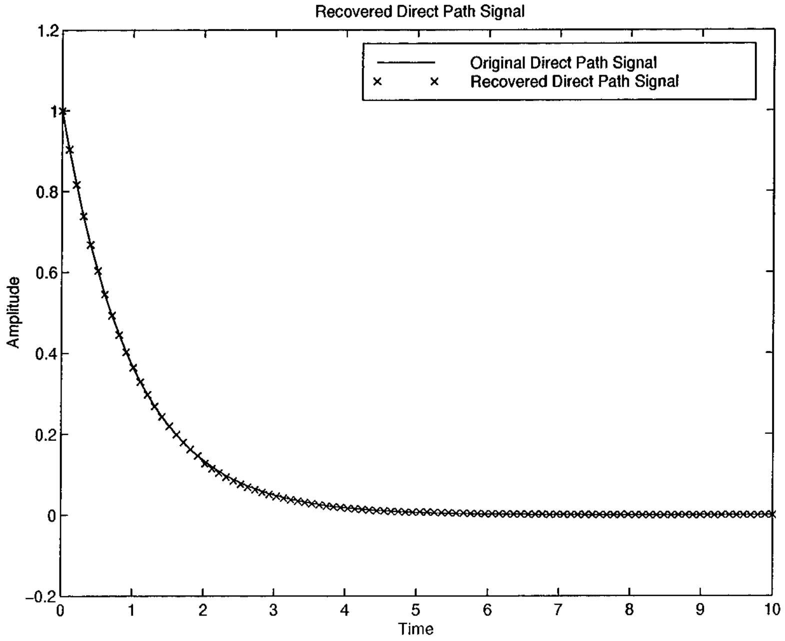

Figure 9 Recovered Direct Path Signal for Example Signal

III. Complex Cepstrum Adaptive Filter

3.1 Overview

This chapter describes the Complex Cepstrum Adaptive Filter (CCAF) and the non-coherent delay lock loop, also commonly called an early-late gate. Additionally, the chapter describes the input signal model. The first section covers the signal model. The subsequent sections discuss the CCAF and it's components. The final section describes the NCDLL.

3.2 Received Signal Model

Prior to processing the received signal for multipath mitigation, it is assumed that the signal has been mixed down to some convenient intermediate frequency (IF). If the received multipath signal is assumed to have one reflection, the received signal can be described by the following equation:

$$s(t) = \sqrt{2P}\,a_0\,c(t - \tau_d)\cos(\omega_{IF}t + \theta_0) + \sqrt{2P}\,a_1\,c(t - \tau_m)\cos(\omega_{IF}t + \theta_1) + n(t) \tag{44}$$

where a_0, a_1 are the direct path and multipath amplitudes, respectively, $\theta_0 = -\omega_0\tau_d$, $\theta_1 = -\omega_0\tau_m$ are the direct path and multipath phases, respectively, τ_d and τ_m are the direct path and multipath propagation delays, respectively, ω_0 is the carrier angular frequency, and $n(t)$ is additive white Gaussian noise (AWGN). Both phases are measured relative to the phase of the transmitter at the time of transmission, which is assumed to be zero without loss of generality.

3.3 Complex Cepstrum Adaptive Filter

3.3.1 Overview. A block diagram of the CCAF is shown in Figure 10. This unit uses the complex cepstrum in parallel with the power cepstrum as described in Chapter II. The system calculates the power cepstrum for detection of the peaks associated with the multipath signal. The peak delays are then provided to the adaptive weighted comb filter which removes the corresponding peaks from the complex cepstrum. An inverse complex cepstrum is calculated, yielding a replica of the direct path signal. The fidelity of the direct path replica depends on several factors including signal-to-noise ratio and aliasing effects. These factors will be discussed in the sections that follow. It is proposed that the CCAF be inserted into a typical GPS receiver code tracking loop as shown by the dotted box in Figure 10. By placing the CCAF in this position, multipath interference can be removed from the composite signal prior to code tracking, thus eliminating code tracking error caused by the multipath.

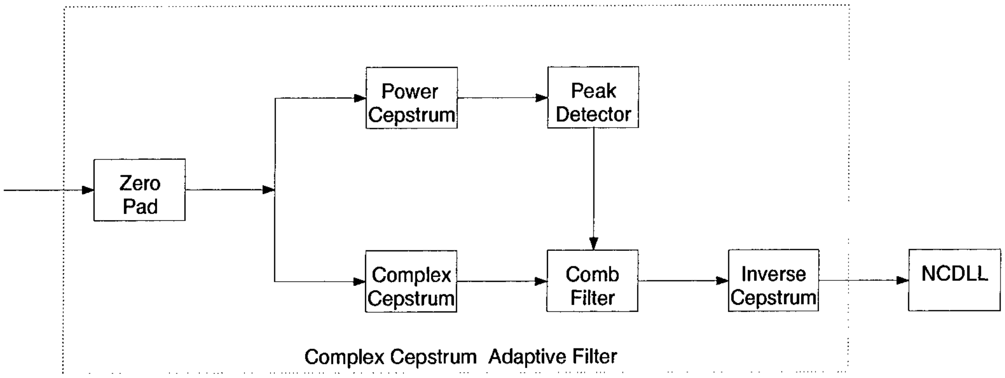

Figure 10 A Code Tracking Loop Employing the Complex Cepstrum Adaptive Filter

3.3.2 Zero Padding. The zero padding block in Figure 10 inputs a block of data (taken to be one period of code for this work), and appends zeros to the end of the data record. This has two effects on the data processing. First, if the correct number of zeros are appended, the record length can be increased to be a power of two. This allows computation of the discrete Fourier transform (DFT) and the inverse DFT using faster computation algorithms, thus speeding up processing time. Second, as discussed in (3), appending zeros increases the frequency domain resolution. This increase reduces or eliminates cepstral (pseudo-time) domain aliasing, and reduces phase unwrapping errors.

3.3.3 Cepstra Calculations. Following zero padding, the power and complex cepstra of the data record are calculated. As mentioned earlier, both cepstra are calculated because it is often easier to detect multipath peaks using the power cepstrum rather than the complex cepstrum (3); however, the complex cepstrum must be used for time domain signal reconstruction following filtering. These calculations are performed using the definitions given in Equations 8 and 10. The calculations are covered extensively in Chapter II, and will not be discussed again here.

3.3.4 Peak Detection. In order to remove the reflection from the complex cepstrum, the first peak associated with the reflected signal must be detected. A magnitude plot of a typical power cepstrum for a GPS multipath signal is shown in Figure 11. The first delta function in the power cepstrum corresponds to the delay of the reflected signal, assuming a single reflection, and provides all the information necessary to filter out the reflection. The peak detector, of Figure 10, performs the function of determining the delay associated with the first peak. In Chapter II, it is shown that the power and complex cepstra contain delta functions at the multipath delay and all integer multiples of the delay. Therefore, the peak detector must be able to detect the first delta function for the single reflection case. To perform this task, the peak detector compares the power cepstrum value at each point to the two adjacent points. A point which has a magnitude exceeding those of the two adjacent points by more than some user set threshold is declared a multipath peak. In Equation 20, it is seen that the first delta function in the power cepstrum will have magnitude $|a_0/\left(1+a_0^2\right)| \leq 1/2$ (see Equation 16) and each subsequent delta will have magnitude $\left[a_0/\left(1+a_0^2\right)\right]^k$ where k is an integer corresponding to the harmonic number of the delta function (ie. $k = 1, 2, 3 \ldots$). Therefore, the first peak will have a greater magnitude than each subsequent peak. The peak detector searches for the first peak present and declares the corresponding delay to be that of the reflection. As shown in Figure 11, the power cepstrum will contain peaks at the chip interval as well as the multipath delay. However, as discussed in the next section, the adaptive weighted comb filter is designed not to filter these peaks. Therefore, the detector is designed not to detect a delay of exactly 1.0 chip.

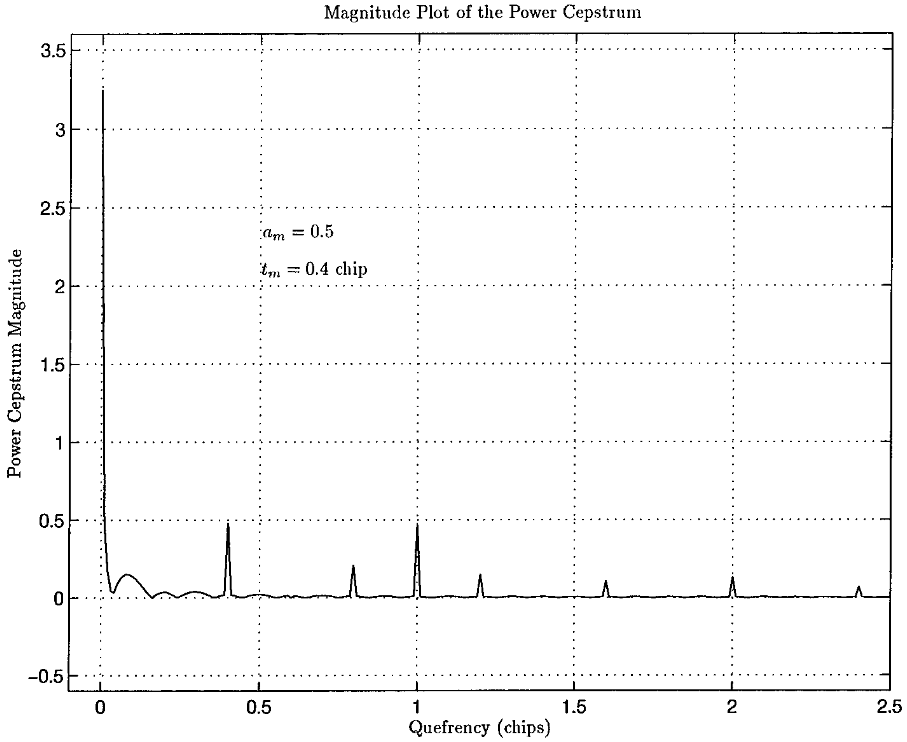

Figure 11 Typical Power Cepstrum Magnitude for a GPS Multipath Composite Signal

3.3.5 Adaptive Weighted Comb Filter. The adaptive weighted comb filter performs the function of removing the reflected signal from the complex cepstrum. Following detection by the peak detector, the delay of the reflection is provided to the adaptive weighted comb filter. This filter then replaces the complex cepstrum data point at the detected delay with the average of the previous point and the next point. Additionally, the comb filter replaces the data points at each integer multiple of the multipath delay with the average of the two adjacent points. In this manner the delta functions in the complex cepstrum, due to the reflected signal, are replaced by linearly interpolated estimates of the complex cepstrum values for the direct path signal. A portion of a typical filtered and unfiltered complex cepstrum is shown in Figure 12. Observe in Figure 12 that the complex cepstrum contains delta functions at the chip interval, as well as those due to the multipath signal. Simulations show that these peaks are important for proper reconstruction of the direct path estimate. Therefore, the adaptive weighted comb filter is designed not to filter the complex cepstrum at the chip interval.

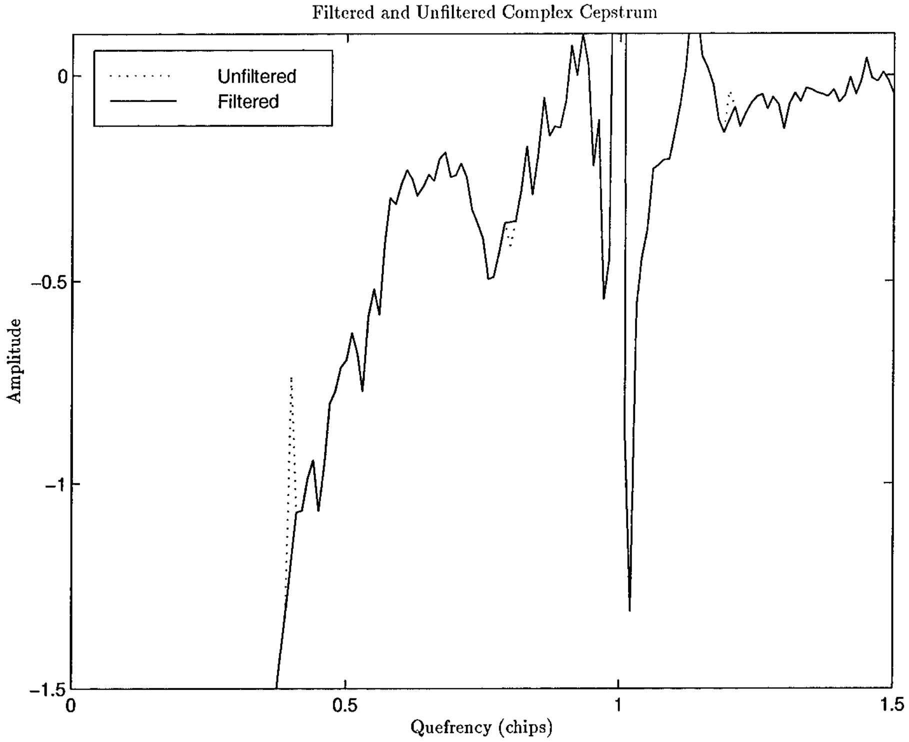

Figure 12 Typical Filtered and Unfiltered Complex Cepstrum for a GPS Multipath Composite Signal

To better understand the adaptive weighted comb filter, consider a complex cepstrum data sequence, $\hat{x}(nT)$, where n is the sample number and T is the sampling interval. Assume that $\hat{x}(nT)$ is N samples long, and contains an initial multipath peak in the c_k element. Then, $\hat{x}(nT)$ could be written as follows:

$$\hat{x}(nT) = \left[\, c_1\; c_2\; \ldots\; c_{k-1}\; c_k\; c_{k+1}\; \ldots\; c_{2k-1}\; c_{2k}\; c_{2k+1}\; \ldots\; c_N \,\right]$$

where c_i is the i^{th} data point in the complex cepstrum.

Next, a filter vector W, also of length N, is defined as follows:

$$W = \left[1\,1\,\ldots\,1\,\frac{c_{k-1}+c_{k+1}}{2\,c_k}\,1\,\ldots\,1\,\frac{c_{2k-1}+c_{2k+1}}{2\,c_{2k}}\,1\,\ldots\,1 \right]$$

where W consists of ones for all elements except those corresponding to the multipath peaks in $\hat{x}(nT)$ (i. e. the elements k, 2k, 3k, etc.). It is easily shown that if the vectors $\hat{x}(nT)$ and W are multiplied in a point-wise fashion, the result is a filtered complex cepstrum vector with the multipath reflections removed.

3.3.6 Inverse Complex Cepstrum. After removing the multipath reflection, a time domain estimate of the direct path signal is recovered through an inverse complex cepstrum operation. The steps of this calculation are shown in Figure 5 of Chapter II. The process is simply the inverse of the complex cepstrum process. First, a forward Fourier transform is calculated, followed by a complex exponentiation (with phase wrapping), and finally an inverse Fourier transform is calculated yielding a time domain signal. The phase wrapping procedure is the inverse of the chosen phase unwrapping procedure used in the complex cepstrum calculation, as discussed in Section 2.2.2.

3.4 The Non-Coherent Delay Lock Loop

Tracking of the GPS spreading code is a primary function of the GPS receiver. This tracking provides a method of determining the line-of-sight distance from the receiver to the satellite, and also acts to despread the received signal. Code tracking is typically accomplished using a delay lock loop (DLL), frequently called an early-late gate. The delay lock loop may operate in either a coherent or non-coherent fashion. Since coherent DLLs require an estimate of carrier phase, considerable interaction between the carrier and code tracking loops is necessary for a coherent DLL to function properly. Additionally, cycle slips in the carrier tracking loop can cause a loss of code tracking lock when a coherent DLL is used. For these reasons, the coherent DLL is considered somewhat fragile and typically not used for GPS applications (9). Therefore, the non-coherent delay lock loop will be presented here. For analysis purposes, a block diagram of a simplified

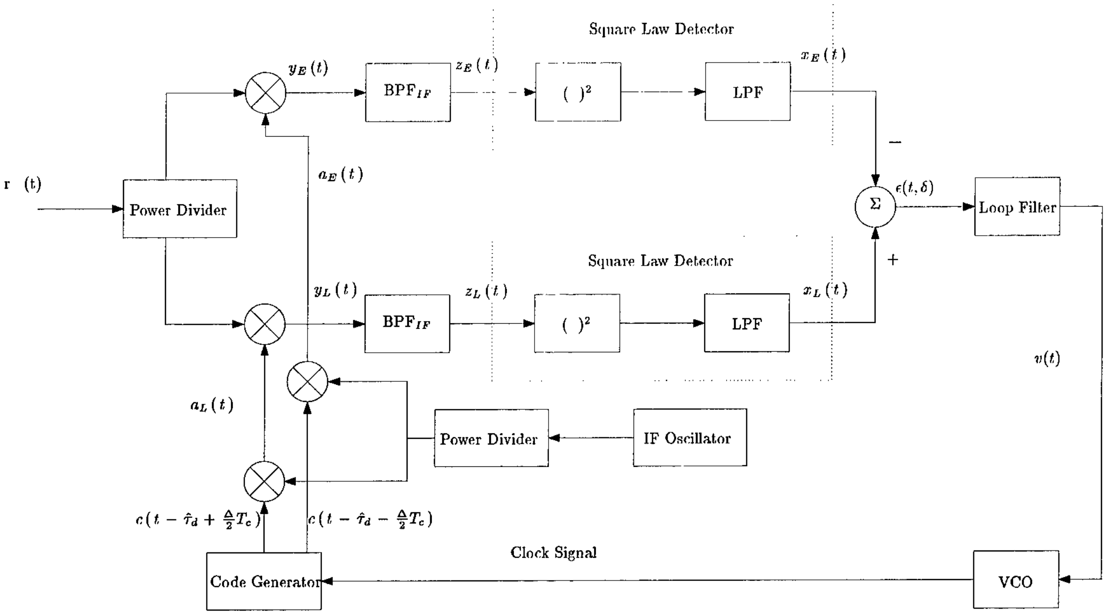

Figure 13 The Non-Coherent Delay Lock Loop

NCDLL is shown in Figure 13. In this DLL, the received signal is cross correlated with early and late versions of the locally generated spreading code replica. The results of these correlations are then bandpass filtered, squared, lowpass filtered, and subtracted to form the discriminator output for the loop. To close the tracking loop, the discriminator output is filtered via a loop filter, then input to a voltage controlled clock which triggers the PN sequence generator producing the on time code replica.

3.4.0.1 Signal Component NCDLL Analysis. The following analysis patterns the development of (10). Consider a DS/SS signal

$$r(t) = \sqrt{2P}\,c(t-\tau_d)\cos[\omega_0 t + \theta_d(t-\tau_d) + \phi] + n(t) \tag{45}$$

where $r(t)$ is the received signal, P is the received power, τ_d is the signal propagation delay, $c(t-\tau_d)$ is the delayed spreading code, ω_0 is the carrier radian frequency, $\theta_d(t-\tau_d)$ is the delayed data signal, ϕ is an arbitrary carrier phase angle, and $n(t)$ is bandlimited zero mean white Gaussian noise, introduced by the channel.

Assuming a two sided noise power spectral density of $N_0/2$ W/Hz, the noise can be represented in terms of its in-phase, $n_I(t)$, and quadrature, $n_Q(t)$, components.

$$n(t) = \sqrt{2}\, n_I(t) \cos(\omega_0 t) - \sqrt{2}\, n_Q(t) \sin(\omega_0 t) \tag{46}$$

After power division, the received signal in each correlator arm becomes

$$r'(t) = \sqrt{P}\, c(t-\tau_d) \cos\left[\omega_0 t + \theta_d(t-\tau_d) + \phi\right] + n_I(t) \cos(\omega_0 t) - n_Q(t) \sin(\omega_0 t) \tag{47}$$

This signal is subsequently correlated with early and late replicas of the spreading code. Assuming that the NCDLL operates at a fixed intermediate frequency, the early and late locally generated signals are given by Equations 48

$$\begin{aligned} a_E(t) &= \sqrt{K_1}\, c\left(t - \hat{\tau}_d + \frac{\Delta}{2} T_c\right) \cos\left[(\omega_0 - \omega_{IF})t + \phi'\right] \\ a_L(t) &= \sqrt{K_1}\, c\left(t - \hat{\tau}_d - \frac{\Delta}{2} T_c\right) \cos\left[(\omega_0 - \omega_{IF})t + \phi'\right] \end{aligned} \tag{48}$$

where a_E and a_L are the early and late correlation signals, respectively, K_1 is an RF-to-IF conversion loss constant, $\hat{\tau}_d$ is the loop's estimate of the propagation delay, Δ is the correlator spacing in chips, and ω_{IF} is an intermediate frequency.

The early correlator output is

$$\begin{aligned} y_E(t) &= r'(t) \cdot a_E(t) \qquad (49) \\ &= \left[\sqrt{P}\, c(t-\tau_d) \cos\left[\omega_0 t + \theta_d(t-\tau_d) + \phi\right] + n_I(t) \cos(\omega_0 t) - n_Q(t) \sin(\omega_0 t)\right] \cdot \\ &\quad \left[\sqrt{K_1}\, c\left(t - \hat{\tau}_d + \frac{\Delta}{2} T_c\right) \cos\left[(\omega_0 - \omega_{IF})t + \phi'\right]\right] \end{aligned}$$

Using appropriate trigonometric identities, the above equation may be simplified by noting that $y_E\,(t)$ is filtered by a bandpass filter centered at the intermediate frequency to obtain $z_E\,(t)$. Therefore, only the difference terms of the simplification need be considered. Thus, the early correlator output, $z_E\,(t)$, becomes

$$\begin{aligned} z_E(t) &= \sqrt{PK_1}\, c(t-\tau_d)\, c(t-\hat{\tau}_d + \frac{\Delta}{2} T_c) \cos\left(\omega_{IF} t + \theta_d(t-\tau_d) + \phi - \phi'\right) \qquad (50) \\ &\quad + \sqrt{K_1}\, c(t-\hat{\tau}_d + \frac{\Delta}{2} T_c) \left[n_I(t) \cos\left(\omega_{IF} t - \phi'\right) - n_q(t) \sin\left(\omega_{IF} t - \phi'\right)\right] \end{aligned}$$

Using a similar process, the late correlator signal, $z_L(t)$, may be expressed as

$$\begin{aligned} z_L(t) &= \sqrt{PK_1}\, c(t-\tau_d)\, c(t-\hat{\tau}_d - \frac{\Delta}{2} T_c) \cos\left(\omega_{IF} t + \theta_d(t-\tau_d) + \phi - \phi'\right) \qquad (51) \\ &\quad + \sqrt{K_1}\, c(t-\hat{\tau}_d - \frac{\Delta}{2} T_c) \left[n_I(t) \cos\left(\omega_{IF} t - \phi'\right) - n_q(t) \sin\left(\omega_{IF} t - \phi'\right)\right] \end{aligned}$$

From the previous two equations, it seen that the signals $z_E\,(t)$ and $z_L\,(t)$ are composed of the desired signal and the noise terms, including code self noise[1]. If the spread spectrum processing gain is sufficiently high (typically greater than 10), the self noise term can be neglected. According to (10), "The amount of performance improvement that is achieved through the use of spread spectrum is defined as the processing gain of the spread spectrum system." The processing gain is

---

[1] Self noise is defined as $sn = E\left[c(t + \tau)\, c'(t + \hat{\tau})\right] - c(t + \tau)\, c'(t + \hat{\tau})$ where E is the expected value, $c(t + \tau)$ is the spreading code, $c'(t + \hat{\tau})$ is the derivative of the spreading code with respect to time, τ is the propagation delay, and $\hat{\tau}$ is the NCDLL estimate of the propagation delay. Self noise is a broadband noise-like waveform (9).

typically defined to be the spread spectrum bandwidth divided by the data bandwidth. Using this definition, the processing gain for GPS C/A code can be calculated as follows

$$\begin{aligned} G_p &= \frac{T}{T_c} \\ &= \frac{\frac{1}{50}}{\frac{1}{1.023 \times 10^6}} \\ &= 20460 \end{aligned} \tag{52}$$

where T is the data bit time, and T_c is the chip width, both in seconds.

From Equation 52, it is seen that the processing gain for GPS C/A code is indeed high, and therefore the code self noise component can be neglected. Ignoring the self noise component and the AWGN, the noise-free components of z_E and z_L are

$$\begin{aligned} z_{sE}(t) &= \sqrt{KP}\, c(t - \tau_d)\, c(t - \hat{\tau}_d + T_c) \cos[\omega_{IF}t + \theta_d(t - \tau_d) + \phi - \phi'] \\ z_{sL}(t) &= \sqrt{KP}\, c(t - \tau_d)\, c(t - \hat{\tau}_d - T_c) \cos[\omega_{IF}t + \theta_d(t - \tau_d) + \phi - \phi'] \end{aligned} \tag{53}$$

where the "s" notation is used to distinguish from the more complete signals of Equations 50 and 51.

The dc component of the spreading code multiplication, by definition, is the code autocorrelation, $R_c(\tau)$, evaluated at $\tau = \tau_d - \hat{\tau}_d + (\Delta/2)\,T_c$ in the case of z_E, and $\tau = \tau_d - \hat{\tau}_d - (\Delta/2)\,T_c$ in the case of z_L; τ_d is the propagation delay, $\hat{\tau}_d$ is an estimate of the propagation delay, and Δ is the correlator spacing in chips. Defining the NCDLL tracking error as $\delta = (\tau_d - \hat{\tau}_d)\,/T_c$,

$$\begin{aligned} z_{sE}(t) &= \sqrt{KP}\, R_c\left[\left(\delta + \frac{\Delta}{2}\right) T_c\right] \cos[\omega_{IF}t + \theta_d(t - \tau_d) + \phi - \phi'] \\ z_{sL}(t) &= \sqrt{KP}\, R_c\left[\left(\delta - \frac{\Delta}{2}\right) T_c\right] \cos[\omega_{IF}t + \theta_d(t - \tau_d) + \phi - \phi'] \end{aligned} \tag{54}$$

Assuming the bandpass filters are designed to pass IF signals with no distortion of the messages, the input to the square-law envelope detectors is given by Equation 54 above. Also, assuming that the low pass filters completely block signal components at $2\,\omega_{IF}$, the signal component of the NCDLL discriminator can be shown to be (10)

$$\begin{aligned} \epsilon(t,\delta) &= \left[x_L^2(t) - x_E^2(t)\right] \\ &= \frac{1}{2}K_1\,P\left\{R_c^2\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right] - R_c^2\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right]\right\} \end{aligned} \tag{55}$$

Defining the NCDLL "S-Curve," $S_\Delta(\delta)$, as

$$S_\Delta(\delta) = R_c^2\left[\left(\delta-\frac{\Delta}{2}\right)T_c\right] - R_c^2\left[\left(\delta+\frac{\Delta}{2}\right)T_c\right] \tag{56}$$

which represents the loop's tracking curve. The expected loop discriminator output is consequently

$$\epsilon\,(t,\delta) = \frac{1}{2}K_1\,P\,S_\Delta(\delta) \tag{57}$$

where δ is the NCDLL tracking error.

Recall that Equation 4 is the autocorrelation function for a maximal length PRN sequence. Upon substitution of Equation 4 into Equation 56, and after some simplification (10), the S-Curve is given as

$$S_\Delta(\delta) = \begin{cases} 0 & \text{for} -N + 1 + \frac{\Delta}{2} < \delta \leq -\left(1 + \frac{\Delta}{2}\right) \\ \frac{1}{N^2} - \left[1 + \left(1 + \frac{1}{N}\right)\left(\delta + \frac{\Delta}{2}\right)\right]^2 & \text{for} -\left(1 + \frac{\Delta}{2}\right) < \delta \leq -\frac{\Delta}{2} \\ \frac{1}{N^2} - \left[1 - \left(1 + \frac{1}{N}\right)\left(\delta + \frac{\Delta}{2}\right)\right]^2 & \text{for} -\frac{\Delta}{2} < \delta \leq -\left(1 - \frac{\Delta}{2}\right) \\ 2\left(1 + \frac{1}{N}\right)\left[2 - \left(1 + \frac{1}{N}\right)\Delta\right]\delta & \text{for} -\left(1 - \frac{\Delta}{2}\right) < \delta \leq \left(1 - \frac{\Delta}{2}\right) \\ \left[1 + \left(1 + \frac{1}{N}\right)\left(\delta - \frac{\Delta}{2}\right)\right]^2 - \frac{1}{N^2} & \text{for}\ \left(1 - \frac{\Delta}{2}\right) < \delta \leq \frac{\Delta}{2} \\ \left[1 - \left(1 + \frac{1}{N}\right)\left(\delta - \frac{\Delta}{2}\right)\right]^2 - \frac{1}{N^2} & \text{for}\ \frac{\Delta}{2} < \delta \leq 1 + \frac{\Delta}{2} \\ 0 & \text{for}\ 1 + \frac{\Delta}{2} < \delta \leq N - 1 - \frac{\Delta}{2} \end{cases} \qquad (58)$$

for $\Delta \geq 1.0$, or

$$S_\Delta(\delta) = \begin{cases} 0 & \text{for} -N + 1 + \frac{\Delta}{2} < \delta \leq -\left(1 + \frac{\Delta}{2}\right) \\ \frac{1}{N^2} - \left[1 + \left(\delta + \frac{\Delta}{2}\right)\left(1 + \frac{1}{N}\right)\right]^2 & \text{for} -\left(1 + \frac{\Delta}{2}\right) < \delta \leq \left(\frac{\Delta}{2} - 1\right) \\ -2\left(1 + \frac{1}{N}\right)\Delta\left[1 + \left(1 + \frac{1}{N}\right)\delta\right] & \text{for} -\left(\frac{\Delta}{2} - 1\right) < \delta \leq -\frac{\Delta}{2} \\ 2\left(1 + \frac{1}{N}\right)\left[2 - \left(1 + \frac{1}{N}\right)\Delta\right]\delta & \text{for} -\frac{\Delta}{2} < \delta \leq \frac{\Delta}{2} \\ 2\left(1 + \frac{1}{N}\right)\Delta\left[1 - \left(1 + \frac{1}{N}\right)\delta\right] & \text{for}\ \frac{\Delta}{2} < \delta \leq \left(1 - \frac{\Delta}{2}\right) \\ \left[1 - \left(1 + \frac{1}{N}\right)\left(\delta - \frac{\Delta}{2}\right)\right]^2 - \frac{1}{N^2} & \text{for}\ \left(1 - \frac{\Delta}{2}\right) < \delta \leq \left(1 + \frac{\Delta}{2}\right) \\ 0 & \text{for}\ 1 + \frac{\Delta}{2} < \delta \leq N - 1 - \frac{\Delta}{2} \end{cases} \qquad (59)$$

for $\Delta < 1.0$.

Recall from Chapter I that the autocorrelation function of a periodic maximal length spreading code of period N is also periodic with period N. Therefore, the S-Curve defined above is periodic with period N. Equations 58 and 59 give principal function values which repeat every N chips. Plots of Equations 58 and 59 are shown in Figures 14 and 15, respectively.

It is important to note, as evident in Figures 14 and 15, that the slope of the S-curve near $\delta = 0$ is dependent upon the correlator spacing, Δ. This slope is zero for a correlator spacing of

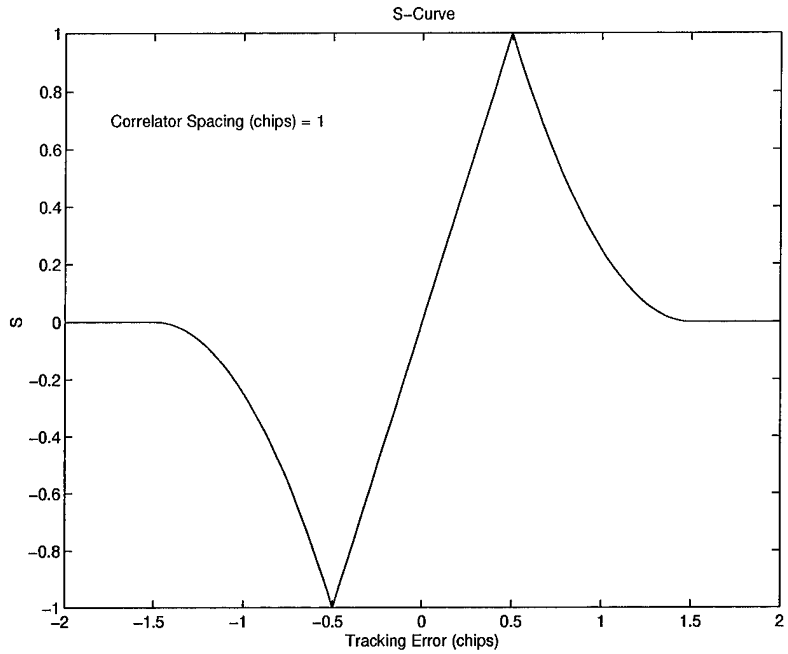

Figure 14 The Non-Coherent Delay Lock Loop S-Curve for $\Delta = 1$ Chip

2 chips; therefore, a correlator spacing of less than 2 chips is always used. Additionally, note from Figures 14 and 15 that the size of linear tracking region about $\delta = 0$ decreases with decreasing Δ. A larger linear tracking region aids in code acquisition (prior to code lock and tracking) because the tracking loop is able to tolerate larger tracking errors while establishing code lock. For this reason, many receivers employ NCDLL's with variable correlator spacing. During code acquisition, the correlator spacing is set at more than 1 chip. Then, once code lock is achieved, the correlator spacing is decreased to some smaller value (usually 1 chip for a typical receiver or 0.1 chip for a narrow correlator receiver).

3.4.0.2 Additive White Gaussian Noise NCDLL Analysis. Having determined the signal component of the discriminator output, next consider the component due to noise. Recall that the noise components at the correlator outputs were given in Equations 50 and 51. Referring

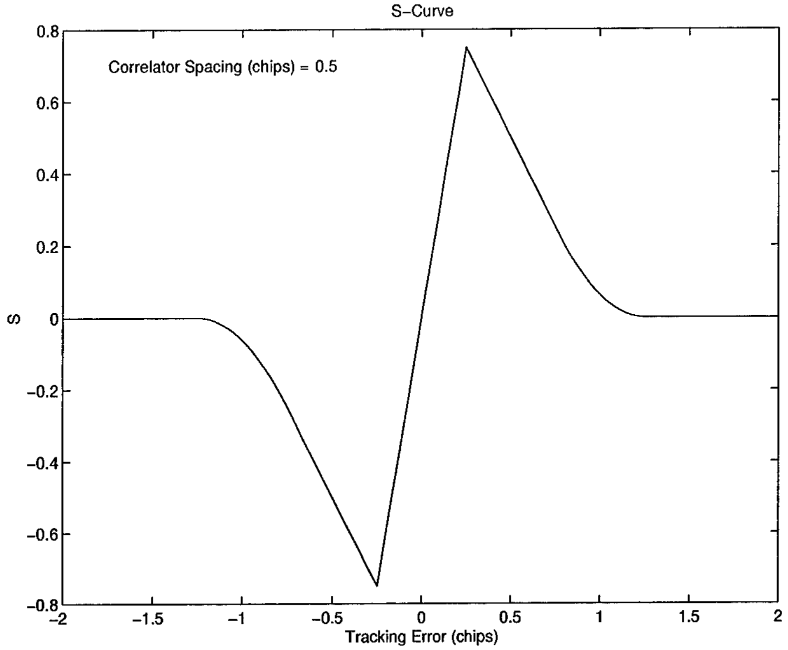

Figure 15 The Non-Coherent Delay Lock Loop S-Curve for $\Delta = 0.5$ Chip

back to those equations, define the noise as follows.

$$n_{E,in}(t) = \sqrt{\frac{K_1}{2}}\, c\left(t - \hat{\tau}_d + \frac{\Delta}{2}T_c\right) n'(t) \tag{60}$$

$$n_{L,in}(t) = \sqrt{\frac{K_1}{2}}\, c\left(t - \hat{\tau}_d - \frac{\Delta}{2}T_c\right) n'(t) \tag{61}$$

where the "E" and "L" subscripts denote the early and late correlator arms, respectively, the "in" subscript denotes noise at the NCDLL input, and

$$n'(t) = \sqrt{2}\, n_I(t) \cos(\omega_{IF}t - \phi') - \sqrt{2}\, n_Q(t) \sin(\omega_{IF}t - \phi') \tag{62}$$

Next, to find the noise component at the bandpass filter output, the noise power spectral density at the input is needed. Let $S_{n'}$ and S_c be the power spectral densities of the noise process, $n'(t)$, and the spreading code, $c(t)$, respectively, and $*$ represent the convolution operator. Since the noise and the spreading code are assumed to be independent, the power spectral densities of $n_{E,in}(t)$ and

$n_{L,in}(t)$ are found by convolving the spreading code power spectral density with that of $n'(t)$

$$S_{n_{j,in}}(\omega) = \tfrac{K_1}{4\pi} S_{n'}(\omega) * S_c(\omega) \quad j = E, L \tag{63}$$

where j designates either the early or late arm of the NCDLL.

Since the spreading code has a large bandwidth, the effect of the convolution of Equation 63 is to spread the noise power over a bandwidth wider than the original. The bandpass filter has a smaller bandwidth with respect to the spreading code and only passes a small fraction of the (spread) noise power spectrum. Consequently, the power spectrum in Equation 63 only needs to be evaluated at frequencies near the intermediate frequency, ω_{IF} (10). Thus, Equation 63 can be approximated by

$$\begin{aligned} S_{n_{j,in}}(\omega) &\approx S_{n_{j,in}}(\omega_{IF}) \\ S_{n_{j,in}}(\omega) &= \begin{cases} \frac{K_1}{2}\left(\frac{N_0}{2}\right) & |\omega \pm \omega_{IF}| < \pi B_N \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{64}$$

where B_N is the one sided bandwidth of the noise process, $K_1/2$ accounts for the RF-IF conversion loss and the power division, and $N_0/2$ is the power spectral density magnitude of $n(t)$, for all ω.

With the approximate noise power spectrum given in Equation 64, the output of the bandpass filters can be calculated using the relationship

$$S_{out}(\omega) = S_{in}(\omega)\,|H(\omega)|^2 \tag{65}$$

where $S_{out}(\omega)$ and $S_{in}(\omega)$ are the power spectra at the output and input, respectively, of a filter with transfer function $H(\omega)$. Assuming that the bandpass filter is an ideal filter with noise bandwidth

B_N yields the following power spectrum for the noise at the filter output.

$$\begin{aligned} S_{n_{j,out}}(\omega) &= \tfrac{K_1}{2}\left(\tfrac{N_0}{2}\right)|1|^2 \quad \text{for} \quad |\omega \pm \omega_{IF}| \leq \pi B_N \\ &= \tfrac{1}{4} K_1 N_0 \qquad\quad\; \text{for} \quad |\omega \pm \omega_{IF}| \leq \pi B_N \end{aligned} \tag{66}$$

With the noise at the bandpass filter output thus characterized, the discriminator output due to the signal and the noise can be calculated as

$$\epsilon(t,\delta) = [z_{sL}(t) + z_{nL}(t)]_{lp}^2 - [z_{sE}(t) + z_{nE}(t)]_{lp}^2 \tag{67}$$

where $z_{sL}(t)$ and $z_{sE}(t)$ are the signal components at the output of the late and early arm bandpass filters, respectively, z_{nL} and z_{nE} are the corresponding noise components, and the lp denotes the lowpass component of the signals.

It can be shown (10) that the discriminator output of Equation 67 is equivalently

$$\begin{aligned} \epsilon(t,\delta) = {} & \frac{1}{2} K_1 P \left\{ R_c^2 \left[\left(\delta - \frac{\Delta}{2} \right) T_c \right] - R_c^2 \left[\left(\delta + \frac{\Delta}{2} \right) T_c \right] \right\} \\ & + \sqrt{2 K_1 P} \left\{ R_c^2 \left[\left(\delta - \frac{\Delta}{2} \right) T_c \right] n'_{L_{bp}I}(t) - R_c^2 \left[\left(\delta + \frac{\Delta}{2} \right) T_c \right] n'_{E_{bp}I}(t) \right\} \\ & \times \cos[\phi - \phi' + \theta_d(t - T_d)] \\ & + \sqrt{2 K_1 P} \left\{ R_c^2 \left[\left(\delta - \frac{\Delta}{2} \right) T_c \right] n'_{L_{bp}Q}(t) - R_c^2 \left[\left(\delta + \frac{\Delta}{2} \right) T_c \right] n'_{E_{bp}Q}(t) \right\} \\ & \times \sin[\phi - \phi' + \theta_d(t - T_d)] \\ & \left[n_{L_{bp}I} \right]^2 + \left[n_{L_{bp}Q} \right]^2 - \left[n_{E_{bp}I} \right]^2 - \left[n_{E_{bp}Q} \right]^2 \end{aligned} \tag{68}$$

where $n_{E_{bp}I}$, $n_{E_{bp}Q}$, $n_{L_{bp}I}$, and $n_{L_{bp}Q}$ are the in-phase and quadrature components of the noise signals in the early and late correlator arms, respectively.

If the noise components of the discriminator output are denoted by $n_\epsilon(t)$, the discriminator of Equation 68 can be written more succinctly as

$$\epsilon(t,\delta) = \frac{1}{2}K_1 P S_\Delta(\delta) + n_\epsilon(t) \tag{69}$$

where $S_\Delta(\delta)$ is the loop S-Curve defined in Equation 56 and δ was previously defined to be the normalized tracking error, $\delta = (\tau_d - \hat{\tau}_d)/T_c$.

Assuming no data modulation is present, the NCDLL can be represented (10) by the equivalent model shown in Figure 16. Although it is unrealistic to assume no data modulation exists, this assumption leads to a simpler model. Additionally, this assumption leads to a worst case performance model in the sense that this model will have the maximum possible noise component at the discriminator output. Therefore, this model is valuable for analyzing the performance of the NCDLL.

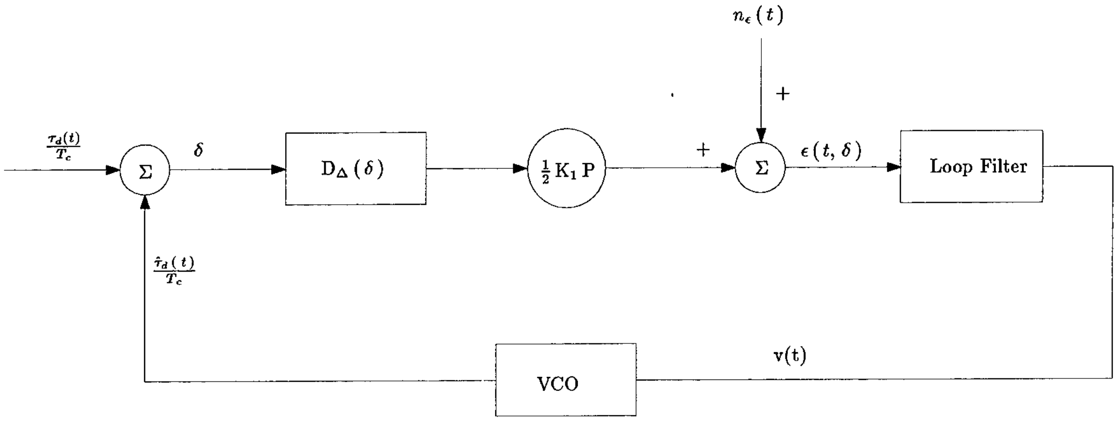

Figure 16 The Non-Coherent Delay Lock Loop Non-Linear Model

In Figure 16, the voltage controlled oscillator (VCO) is described by the following input/output relationship.

$$\frac{\hat{\tau}_d(t)}{T_c} = g_c \int_0^t v(\lambda)\, d\lambda \tag{70}$$

where g_c is the VCO gain with units of Hz/V. Noting from Figure 16 that $v(\lambda)$ is the loop filter output and that the loop filter input is the discriminator, Equation 70 can be re-written as

$$\frac{\hat{\tau}_d(t)}{T_c} = g_c \int_0^t \int_{-\infty}^{\lambda} \epsilon(\alpha,\delta)\, f(\lambda - \alpha)\, d\alpha\, d\lambda \tag{71}$$

where $v(\lambda)$ from Equation 70 has simply been replaced by the convolution of the discriminator output, $\epsilon\,(t,\delta)$, and the loop filter impulse response, $f(t)$.

Substituting $\epsilon(t,\delta)$ from Equation 69 yields

$$\frac{\hat{\tau}_d(t)}{T_c} = g_c \int_0^t \int_{-\infty}^{\lambda} \left[\frac{1}{2}K_1 P S_\Delta\,(\delta(\alpha)) + n_\epsilon(\alpha)\right] f(\lambda - \alpha)\, d\alpha\, d\lambda \tag{72}$$

Equation 72 can be linearized if the tracking error, δ, is small. Under this assumption, the non-linear S-curve can be replaced with the following linear approximation (10)

$$S_\Delta(\delta) \approx 4\left(1 + \frac{1}{N}\right)\left[1 - \left(1 + \frac{1}{N}\right)\frac{\Delta}{2}\right]\delta \tag{73}$$

where Equation 73 comes from Equation 56 evaluated in the region about $\delta = 0$.

Replacing the non-linear element $S_\Delta(\delta)$ in Figure 16 with the approximation of Equation 73 yields the linear model of Figure 17. In this model, the S-curve element, $S_\Delta(\delta)$, and the gain element, $\frac{1}{2}K_1 P$, from Figure 16 are replaced by a single equivalent gain element, K_d, where

$$K_d = 4\left(1 + \frac{1}{N}\right)\left[1 - \left(1 + \frac{1}{N}\right)\frac{\Delta}{2}\right]\left(\frac{1}{2}K_1 P\right) \tag{74}$$

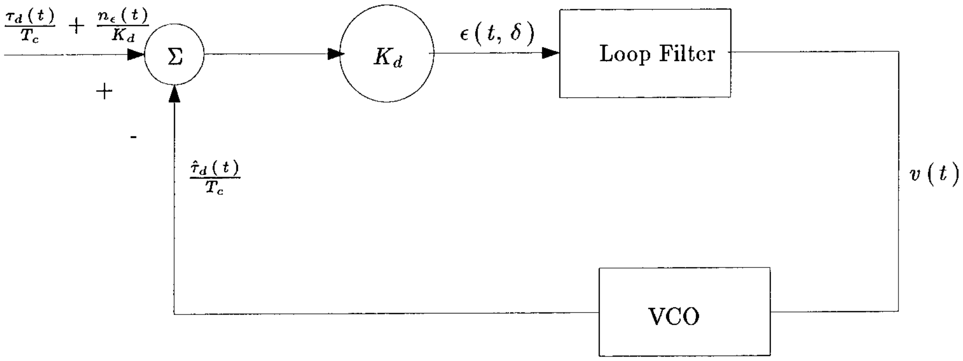
$\frac{\tau_d(t)}{T_c} + \frac{n_\epsilon(t)}{K_d}$
+
-
Σ
K_d
$\epsilon(t, \delta)$
Loop Filter
$v(t)$
VCO
$\frac{\hat{\tau}_d(t)}{T_c}$

Figure 17 The Non-Coherent Delay Lock Loop Linear Model

Let F(s) denote the Laplace transform of f(t). Using the linear model of Figure 17, the closed loop transfer function, H(s), for the NCDLL is

$$H(s) = \frac{K_d\, g_c\, F(s)}{s + K_d\, g_c\, F(s)} = \frac{\hat{\tau}_d(s)}{\tau_d(s)} \tag{75}$$

Additionally, the Laplace transform of the tracking error can be shown (10) to be

$$\delta(s) = \frac{\tau_d(s)}{T_c}\left[\frac{s}{s + K_d\, g_c\, F(s)}\right] \tag{76}$$

Finally, the root-mean-square (RMS) tracking jitter for a loop with two-sided noise bandwidth of W_L is given by (10)

$$\sigma_\delta^2 = \frac{\eta}{2\, K_d^2}\, W_L \tag{77}$$

where $\eta/2$ is the discriminator noise component power spectral density, approximately given as (10)

$$\frac{\eta}{2} = \frac{1}{2}(K_1 N_0)^2\, B_N + \frac{1}{2}K_1^2 N_0 P \left\{R_c^2\left[\left(\delta - \frac{\Delta}{2}\right)T_c\right] + R_c^2\left[\left(\delta + \frac{\Delta}{2}\right)T_c\right]\right\} \tag{78}$$

3.5 Conclusion

This chapter describes the complex cepstrum adaptive filter. It begins with the input signal model, which consists of a direct path component, and a delayed and attenuated reflection of the direct path. Next, each component of the CCAF is described. Detail is given concerning the function and design of the CCAF. Finally, an analysis of the non-coherent delay lock loop is presented. The non-linear and linear models of the NCDLL are given. The chapter concludes by presenting the NCDLL transfer function, tracking error, and RMS tracking jitter.

IV. Computer Simulations and Results

4.1 Overview

This chapter presents computer simulations of the theory presented in the previous chapters. The purpose of these simulations is to verify the functionality of the complex cepstrum adaptive filter for multipath mitigation, to characterize the noise performance of the CCAF, and to compare the CCAF performance to that of a narrow correlator non-coherent delay lock loop. Two series of simulations are run as follow:

1. CCAF noise performance characterization. Simulations are run with constant multipath parameters and various SNR levels to determine the noise sensitivity of the CCAF.
2. CCAF performance comparison with a narrow correlator NCDLL. These simulations are run using the CCAF in conjunction with a standard non-coherent delay lock loop, employing a 1.0 chip correlator spacing. The performance of the CCAF is compared with that of a narrow correlator NCDLL employing a 0.1 chip correlator spacing.

All simulations are written for Matlab version 5 and run on the Sun workstations available at the Air Force Institute of Technology or on a stand alone personal computer. Simulation results are analyzed using Matlab. All Matlab function files used for the simulations are included in Appendix A.

4.2 Simulation Parameters

For all simulations, an attempt is made to make all parameters consistent with actual GPS parameters. In cases where actual GPS receiver parameters are unknown, simulation parameters are chosen to be values which could reasonably be expected for a GPS receiver. Additionally, when some freedom of choice is available, parameters are chosen to satisfy certain desirable qualities to enhance simulations (for instance, decreased run times). The subsections that follow discuss the simulation parameters chosen.

4.2.1 DS/SS Spreading Code. Actual GPS C/A spreading code is used for these simulations. The code is generated using the GPS toolbox for Matlab, available from the Electrical and Computer Engineering Department at AFIT. The code period is 1023 chips long (approximately 1 ms). Each chip is divided into 100 samples to achieve a resolution of 0.01 chips. Thus, the sampling frequency for all simulations is 102.3 MHz.

4.2.2 Carrier and Direct Path Signals. Since actual GPS receivers typically operate at an intermediate frequency, the "carrier" frequency for all simulations is an appropriate intermediate frequency. In order to keep vector sizes manageable, thus reducing run times, the intermediate frequency for these simulations is chosen to be 5 times the chipping rate, or 5.115 MHz. For convenience, the carrier is chosen to have a power of 1/2 W at the receiver. Also for convenience, the carrier phase, θ_0, and direct path propagation delay, τ_d, are both chosen to equal zero. Finally, the direct path attenuation, a_0 is chosen to equal 1. The choices of direct path phase, propagation delay, and attenuation factor have the effect of normalizing the multipath parameters to the direct path values. Thus, these simulations could represent any actual values for these direct path parameters if the multipath parameters are scaled by appropriate values.

4.2.3 Multipath Parameters. As mentioned in the previous paragraph, the direct path parameters are chosen such that the multipath signal will be normalized to the direct path. As such, the multipath attenuation factor, a_1 is chosen to vary between zero and one. For most simulations, $0.2 \leq a_1 \leq 0.8$ is used. Additionally, the multipath delay, τ_m is varied between approximately 10 nanosecond and 1.5 microseconds. These delays correspond to a normalized delay (in chips) of $0.1 \leq \alpha \leq 1.5$.

4.2.4 NCDLL Design. The non-coherent delay lock loop used for these simulations is an open-loop design. This open-loop design uses the NCDLL S-curve, discussed in Chapter III, to characterize the tracking bias present after processing by the CCAF. The CCAF tracking bias, using an NCDLL with standard 1 chip correlator spacing is compared with the tracking bias of a narrow correlator NCDLL, with 0.1 chip spacing, which does not employ CCAF processing. The open loop design is chosen because tracking bias is the primary metric for comparing one multipath mitigation technique to another. Additionally, since this thesis focuses on a signal processing technique used prior to tracking in the NCDLL, and not on processing techniques within the NCDLL itself, the performance of the NCDLL is not in question for this work. Rather, the amount of improvement provided by the pre-processing of the CCAF is the parameter to be studied. Therefore, closed-loop analysis of the NCDLL, to study parameters such as tracking jitter, is not important for this work.

4.2.5 Simulation Parameters Summary. The simulation parameters from the previous subsections are summarized below.

- Spreading Code: Actual GPS C/A code
- Spreading Code Period: 1023 chips
- Sampling Frequency: 102.3 MHz (100 samples/chip)
- Carrier Intermediate Frequency: 5.115 MHz (5 times chipping rate)
- Carrier Received Power: 1/2 W
- Direct Path Phase, θ_0: 0 rad
- Direct Path Propagation Delay, τ_d: 0 sec
- Direct Path Attenuation Factor, a_0: 1
- Multipath Reflection Attenuation Factor, a_1: $0.2 \leq a_1 \leq 0.8$
- Multipath Reflection Normalized Delay, α: $0.1 \leq \alpha \leq 1.5$ chips

- NCDLL Correlator Standard Spacing: 1 chip

- NCDLL Correlator Narrow Spacing: 0.1 chip

4.3 Signal-to-Noise Ratio Analysis

Signal-to-noise ratio is a fundamental concern for all communication systems. SNRs must be sufficiently high to allow receivers to perform the necessary processing to extract the information from the signal. For GPS C/A code, the typical SNR is -14.9 dB at the spectral peak (9). For most communication systems, a negative SNR would prevent communication; however, for spread spectrum systems, such as GPS, the processing gain allows for communication even when negative SNRs exist. In this research, though, it is found that cepstral processing requires extremely high SNR prior to any processing gain being achieved.

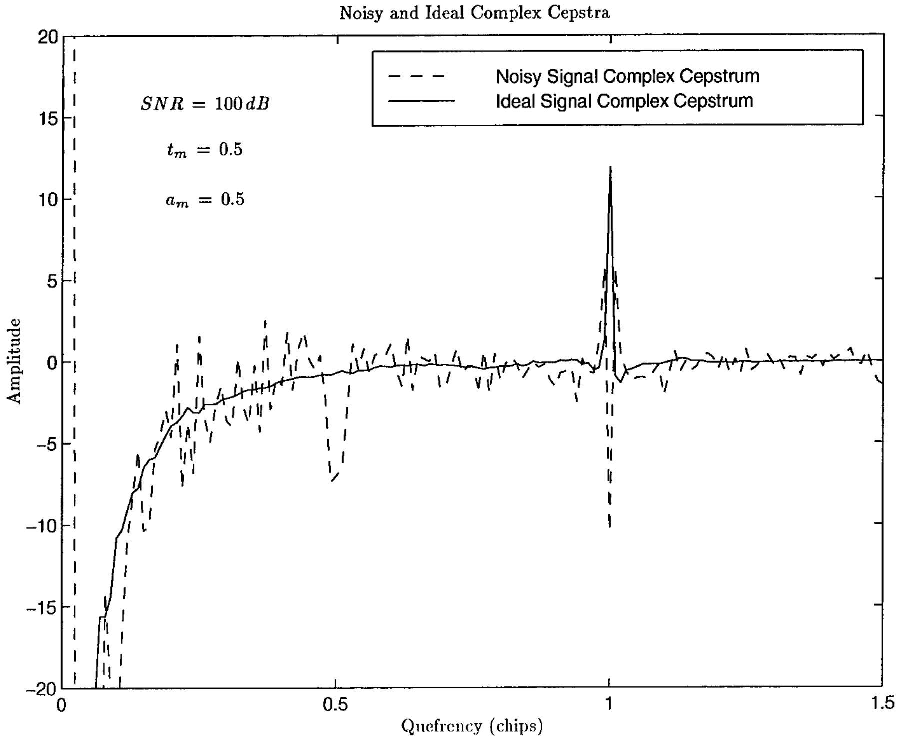

Figure 18 Complex Cepstrum of Noisy and Ideal Signal

For the signal model presented in Chapter III, simulation results show that the SNR must be approximately 225 dB or higher to achieve good signal reconstruction for the direct path estimate. Figure 18 demonstrates the effects of AWGN on the complex cepstrum. In Figure 18 the complex cepstrum of a signal in the absence of AWGN is compared to that for a signal in AWGN with an SNR of 100 dB. It is seen that the noise causes the complex cepstrum to appear very noisy even with such a high input SNR. This random noise in the complex cepstrum causes errors in the multipath filtering process which lead to errors in the recovered signal. Due to the randomness of the data points in the complex cepstrum, the filtered values, which are the average of the adjacent points, also have a certain randomness. This has the effect of causing multipath-like amplitude changes in the recovered signal. Plots demonstrating this effect are shown in Figures 19 and 20 . These amplitude changes greatly degrade or even prevent tracking of the received signal by the code tracking loop. For this reason, the complex cepstrum is found to be applicable for code loop multipath mitigation only under unreasonably high SNR conditions.

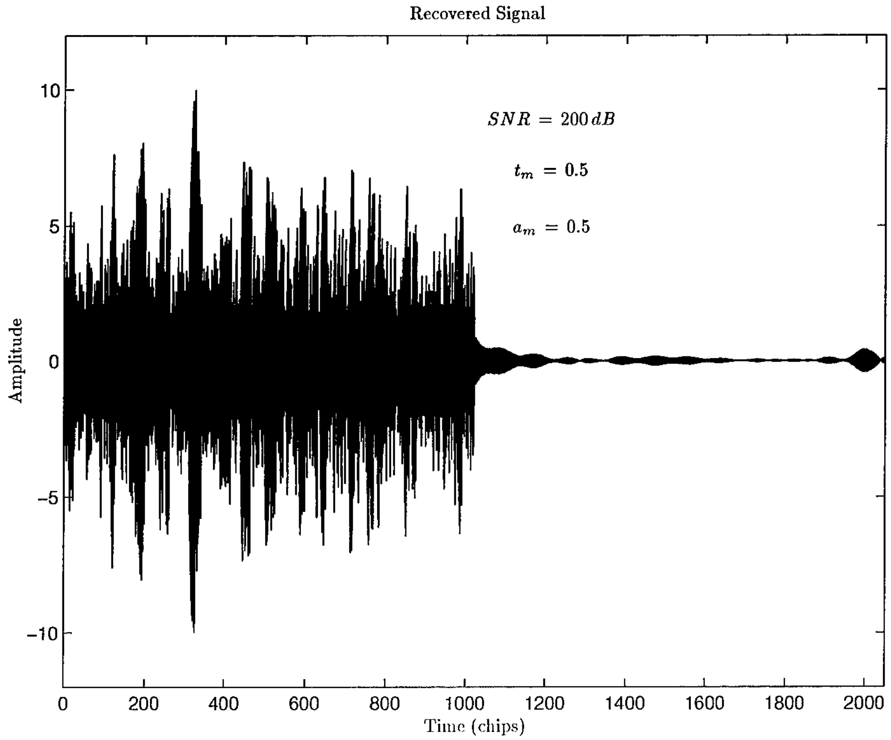

Figure 19 Recovered Signal from a Noisy Complex Cepstrum Process

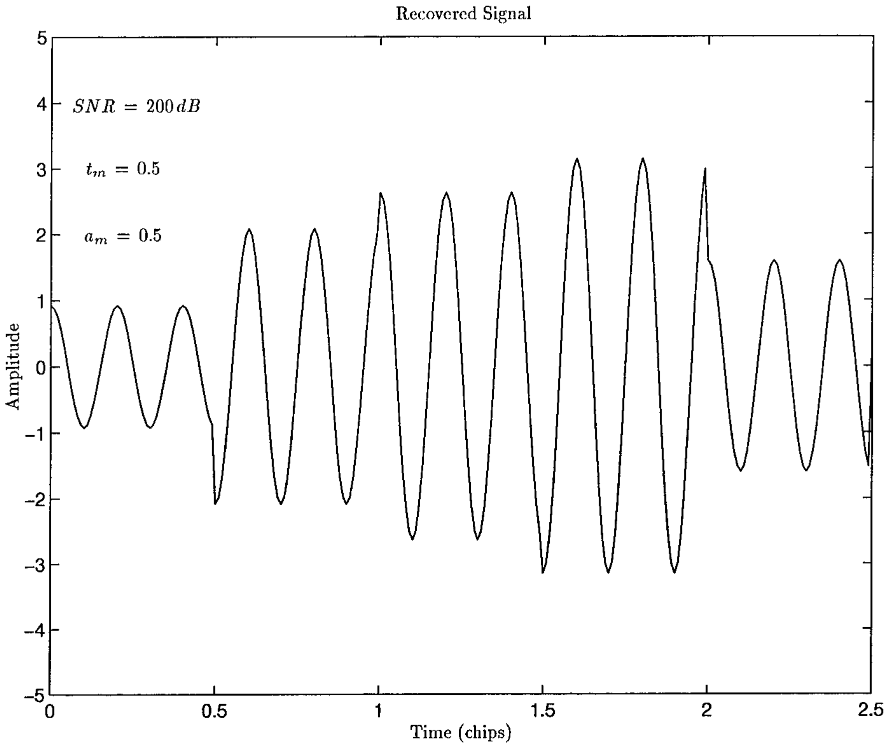

Figure 20 Recovered Signal from a Noisy Complex Cepstrum Process (Expanded Time Scale)

The SNR threshold effect is demonstrated in Figure 21. In this figure, the mean squared amplitude error of the recovered signal is plotted on a semilog scale against the input SNR. Figure 21 shows an exponential increase in mean squared amplitude error for SNRs which are below 160 dB. In this region, the recovered signal is severely degraded, preventing tracking by the NCDLL. However, for good tracking performance the SNR threshold is somewhere near 225 dB, considerably higher than the 160 dB threshold. This is because even very small differences between the recovered signal and the actual direct path signal lead to poor tracking performance by the NCDLL, as compared to a NCDLL tracking the multipath signal. Figure 21 shows that the mean squared amplitude error begins to increase for SNR levels below approximately 225 dB. Although the MSE is still fairly small in the region around 225 dB SNR, it is large enough that the cepstrum processing leads to tracking biases which are larger than those for the same non-coherent delay lock loop tracking the multipath signal without cepstrum processing. For this reason, the input SNR must be above approximately 225 dB to achieve good multipath mitigation using cepstrum processing.

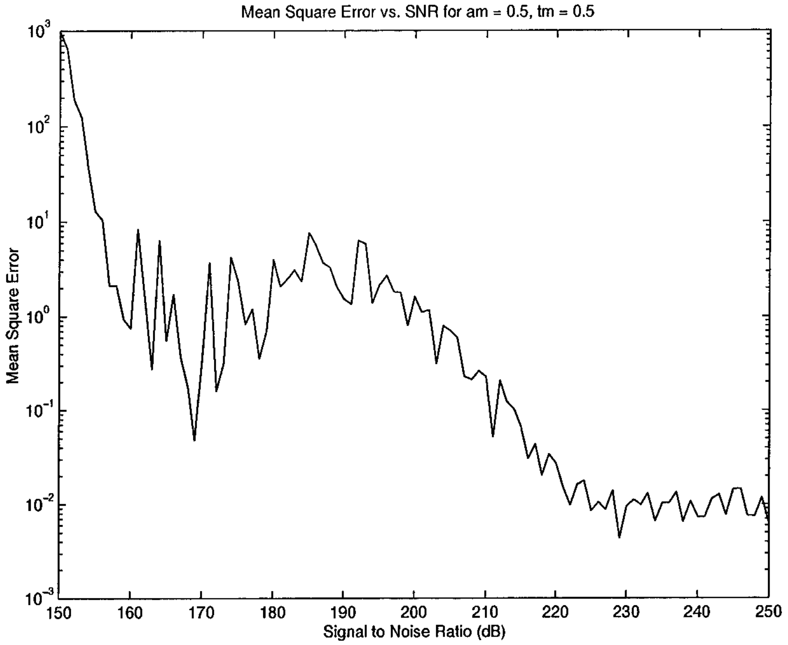

Figure 21 SNR Threshold Effect

To improve noise performance, an input filter would typically be used. However, in this case, filtering the input signal causes severe degradation of the recovered signal. To understand why this is so, refer back to the basic signal model of Equations 29 and 30. Recall that, using these signal models, and following the derivation for calculating the complex and power cepstra, these cepstra are actually time domain deconvolution tools. If the signal is filtered prior to calculating the cepstra, the model of Equations 29 and 30 no longer applies. The new model must account for the fact that the signal has been convolved with the time domain impulse response of the filter. This second convolution (filtering), changes the complex cepstrum, and prevents proper multipath mitigation using the techniques described in Chapters II and III. Figure 22 shows a plot of the complex cepstrum of a multipath signal calculated before and after bandpass filtering the time domain signal to demonstrate this point. The signal in Figure 22 has no noise added so that the effect is due solely to the input filter.

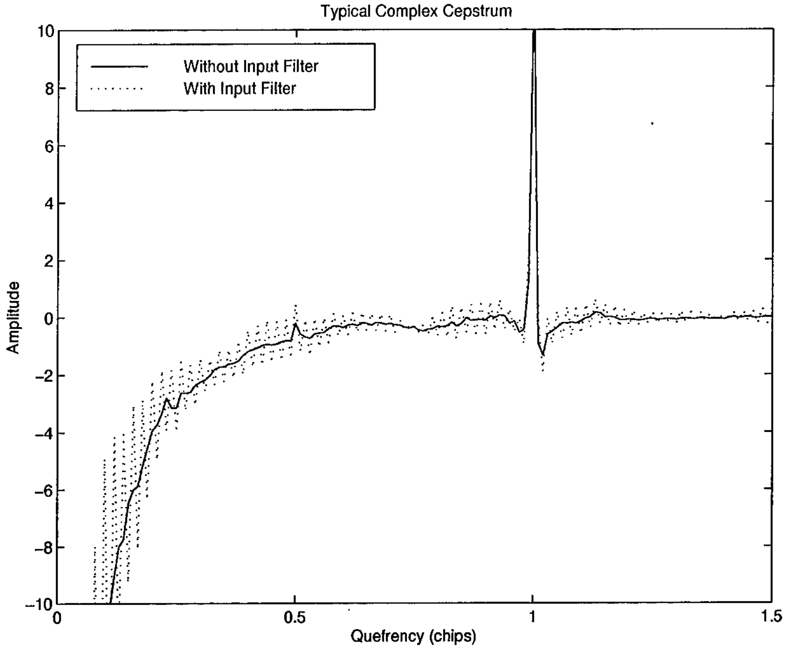

Figure 22 Bandpass Filtering Effect on the Complex Cepstrum

4.4 Cepstrum Processing Analysis

The cepstrum processing for multipath mitigation consists of two steps, as described in Chapter III. First, the power cepstrum is used to detect the multipath signal. Then, the complex cepstrum is used to filter out the multipath. Although both these steps are sensitive to the input SNR, the minimum SNR is set using the complex cepstrum. The power cepstrum is significantly less sensitive to noise effects than the complex cepstrum. This observation will be discussed more in the section that follows.

4.4.1 Multipath Detector. As described in Chapter III, the power cepstrum magnitude can be used to detect the multipath delay by searching for the first delta funtion in the power cepstrum. Simulation shows this process is relatively immune to noise effects assuming the input SNR is about 30 dB or higher, and the threshold for declaring a multipath peak is set appropriately. Magnitude plots of the power cepstrum for typical GPS multipath signals are shown in Figures 23 and 24.

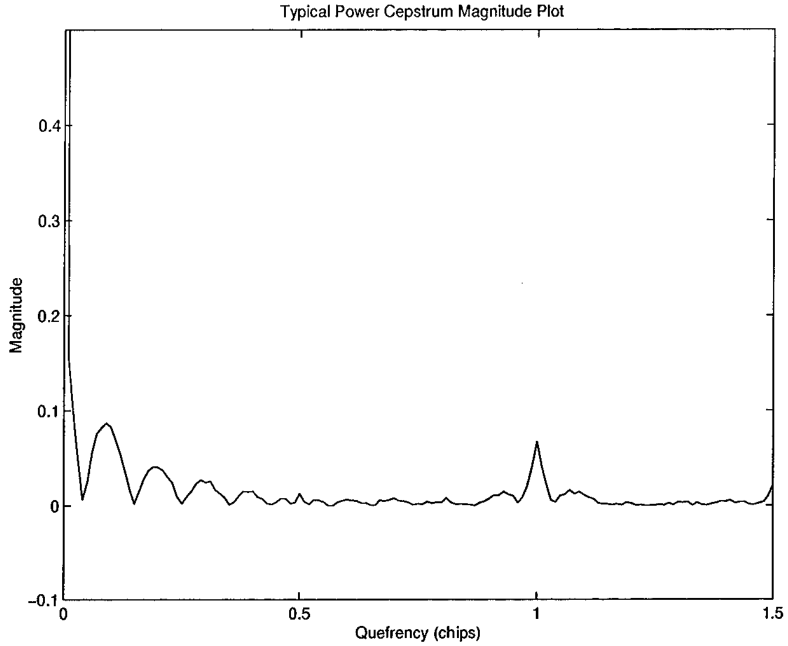

Figure 23 Power Cepstrum for a GPS Multipath Signal, SNR = 30 dB, Multipath Amplitude = 0.5, Multipath Delay = 0.5

Analysis of Figures 23 and 24 reveals two important observations. First, the power cepstrum of a GPS multipath signal contains "rolling hill" type peaks at low quefrencies. These rolling hill peaks, although not caused by multipath, sometimes cause the multipath delay to be incorrectly detected. This effect will be observed later. The second observation is that, as the SNR decreases, the magnitude of the delta functions in the power cepstrum decreases. This decrease drives the detection threshold. As the delta function peaks become smaller, it becomes more difficult to accurately detect the multipath peaks. For the simple detector used in this work, the detection SNR threshold is approximately 30 dB SNR.

To demonstrate that cepstral processing can be used for multipath mitigation, a series of simulations tests the cepstrum system of Chapter III over a variety of multipath scenarios, for SNRs which are above the threshold mentioned earlier in this chapter. For these simulations, the multipath delay is varied from 0.1 chip to 1.5 chips, for multipath amplitudes equal to 0.2, 0.4, 0.6, and 0.8, relative to the direct path amplitude. For each pair of multipath delays and amplitudes,

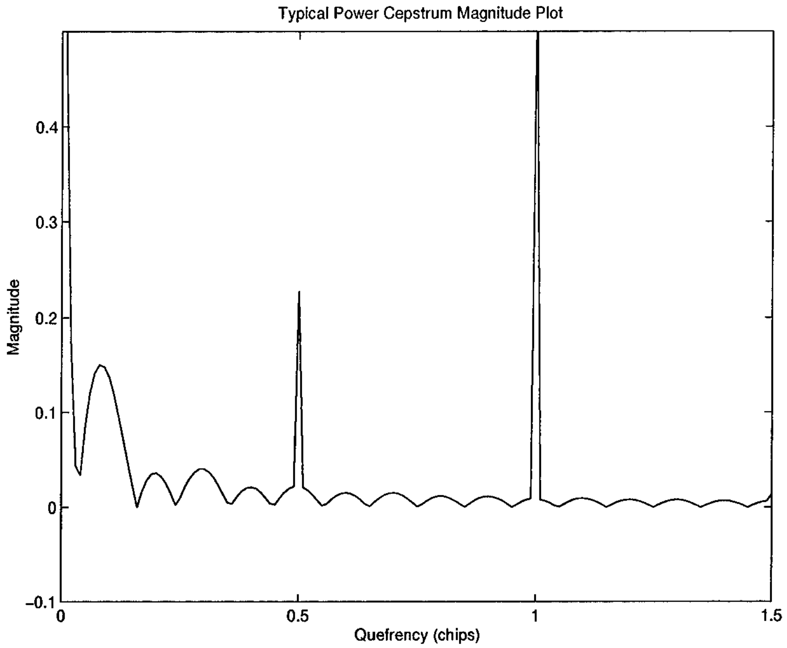

Figure 24 Power Cepstrum for a GPS Multipath Signal, SNR = 250 dB, Multipath Amplitude = 0.5, Multipath Delay = 0.5

10 trials are run adding independent white Gaussian noise to the input signal. Plots of the detected multipath delay (dots) versus the actual multipath delay (line) are shown in Figures 25 through 36. Four observations should be made concerning these plots. First, the detector works well over a wide variety of multipath scenarios. Three points are observed when the multipath detector performance is not ideal, corresponding to delays of 0.1 chip, 0.3 chip, and 1.0 chip. The first two problem spots are due to the rolling hill peaks in the power cepstrum at these low quefrencies. The problem at 1.0 chip is actually the way the detector is designed. As discussed in Chapter III, the power and complex cepstrum both contain natural peaks at the chip interval. Simulations demonstrated that these peaks are important for signal reconstruction, and should not be filtered out. Therefore, since the adaptive weighted comb filter is designed not to filter at the chip interval, the multipath detector is designed not to detect multipath delays of 1.0 chip. This design has the weakness that when a multipath reflection exists at exactly 1.0 chip delay, this cepstrum process cannot remove the multipath. This weakness will be demonstrated in the next section. Finally, when the multipath

reflection delay is incorrectly detected, the detected delay is always 2.0 chips. This is due to the first harmonic of the natural peak at 1.0 chip delay being detected as the multipath signal, and is due to the fact that the detector is designed with the assumption that a multipath reflection exists at some delay.

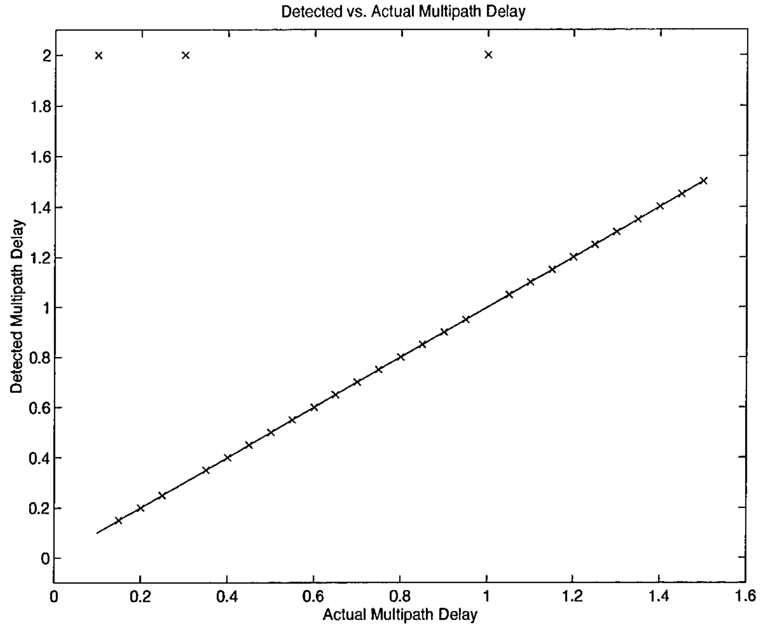

Figure 25 Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.2

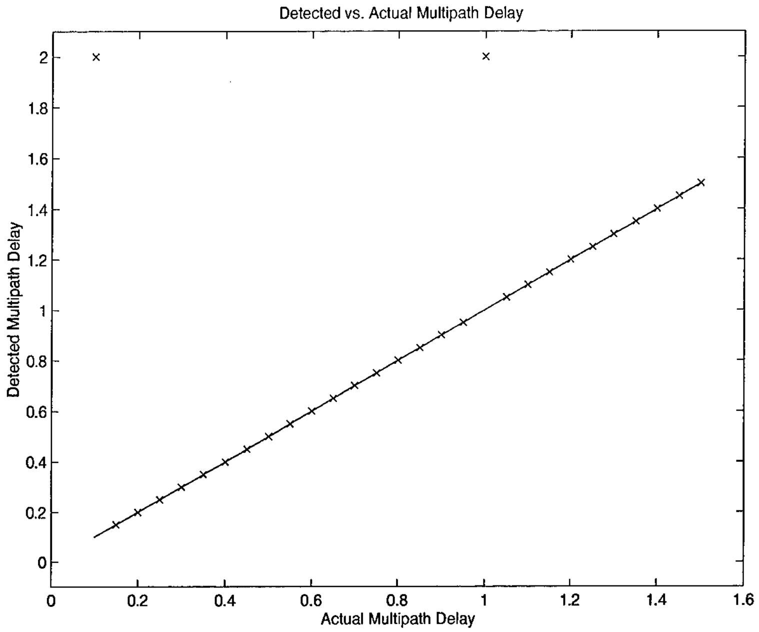

Figure 26 Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.4

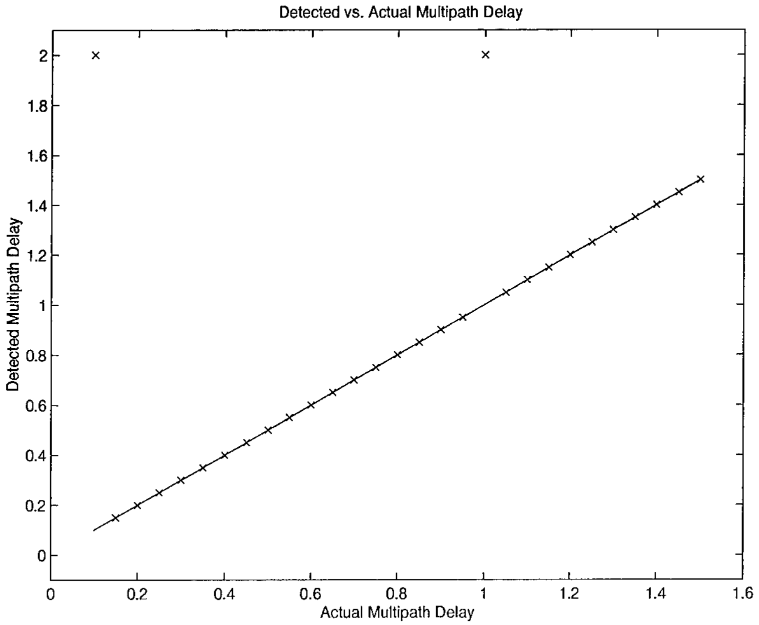

Figure 27 Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.6

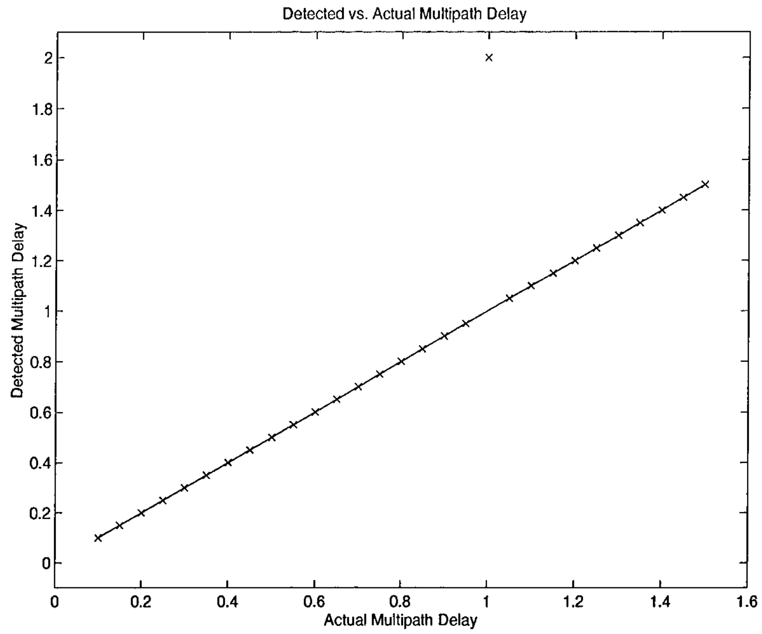

Figure 28 Multipath Detector Test Results for SNR = 1000 dB and Multipath Amplitude = 0.8

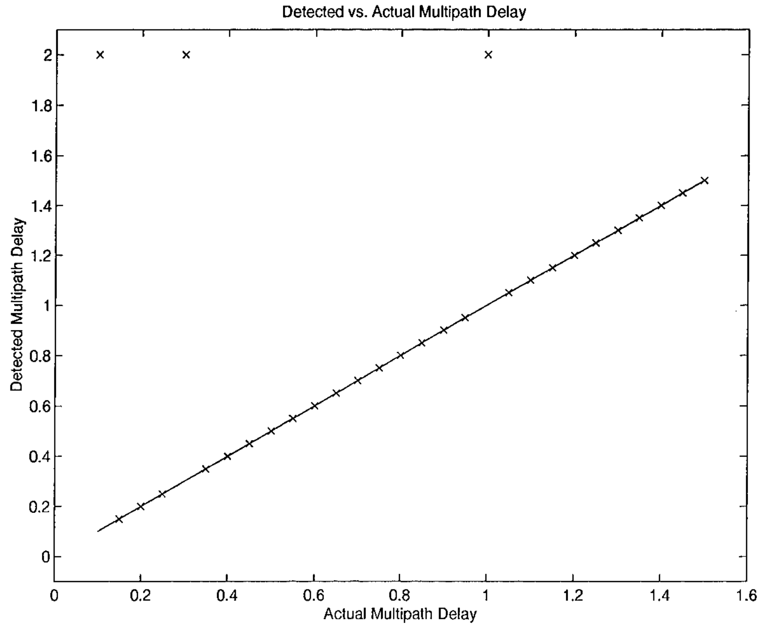

Figure 29 Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.2

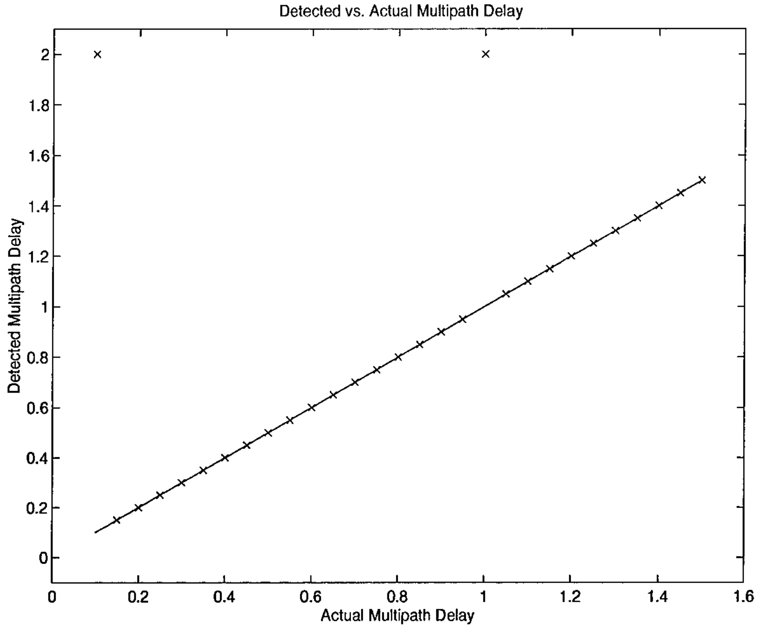

Figure 30 Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.4

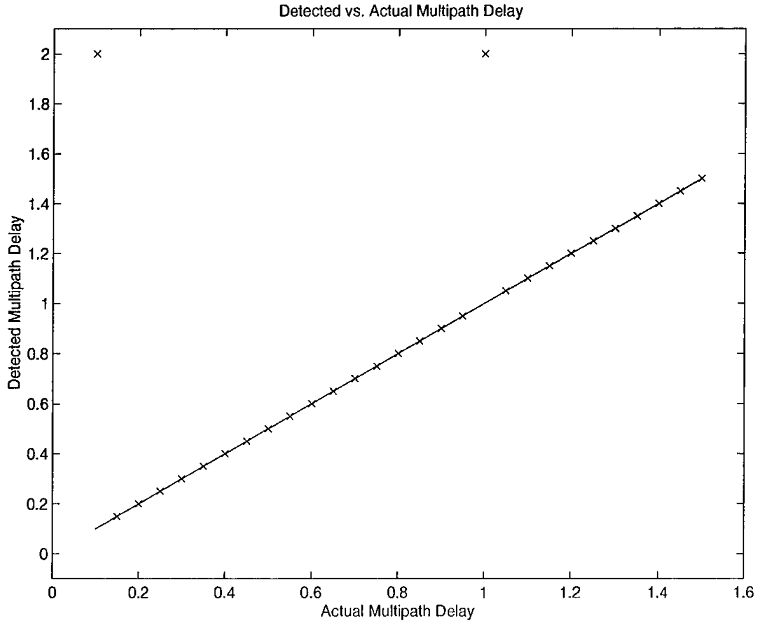

Figure 31 Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.6

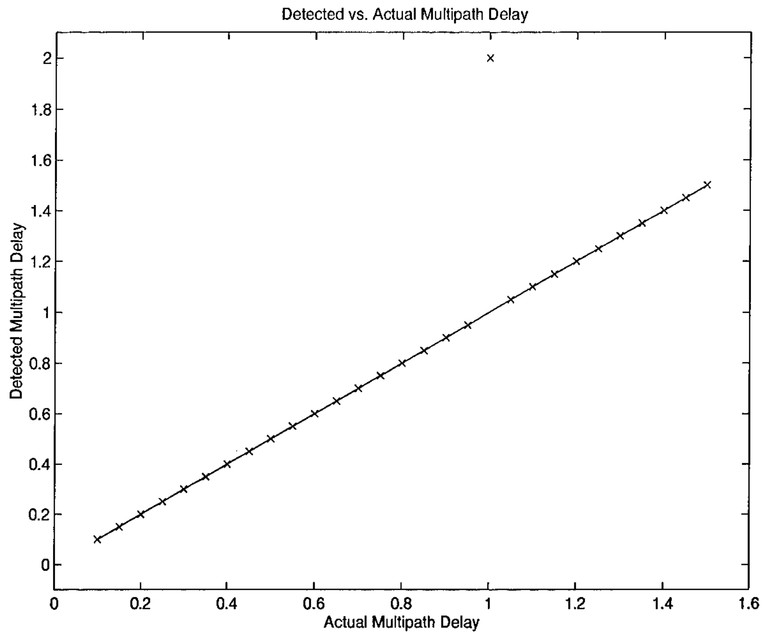

Figure 32 Multipath Detector Test Results for SNR = 250 dB and Multipath Amplitude = 0.8

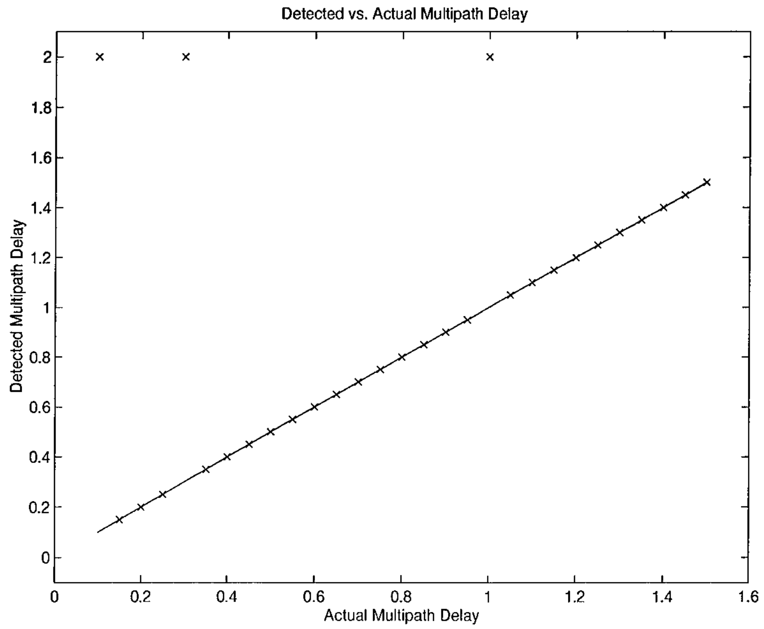

Figure 33 Multipath Detector Test Results for SNR = 223 dB and Multipath Amplitude = 0.2

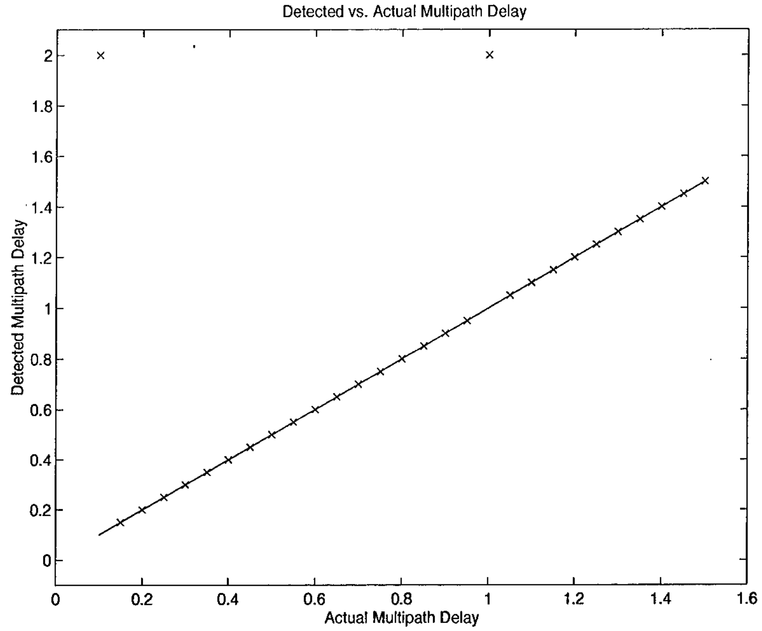

Figure 34 Multipath Detector Test Results for SNR = 223 dB and Multipath Amplitude = 0.4

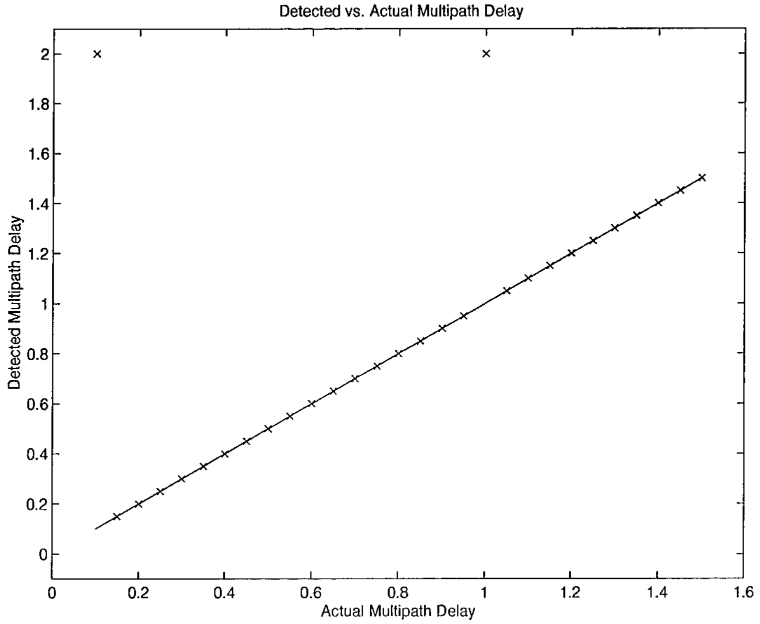

Figure 35 Multipath Detector Test Results for SNR = 223 dB and Multipath Amplitude = 0.6

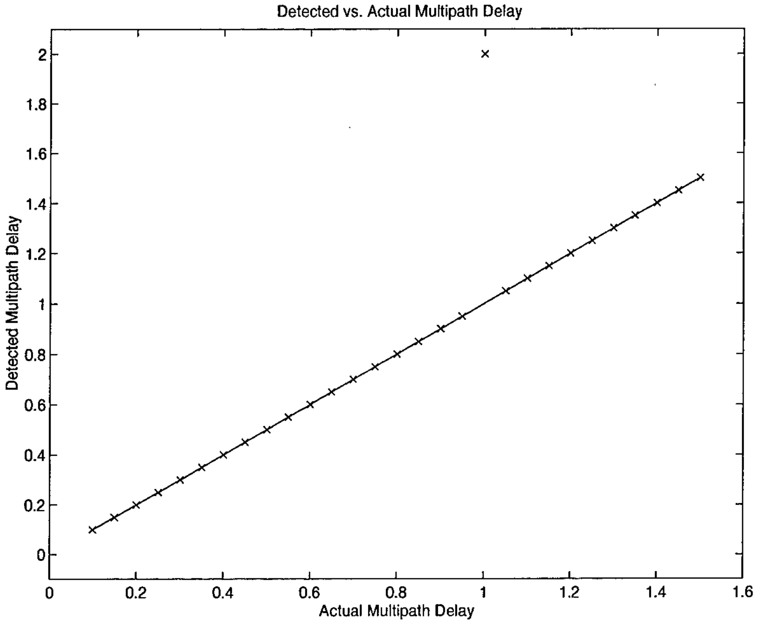

Figure 36 Multipath Detector Test Results for SNR $= 223$ dB and Multipath Amplitude $= 0.8$

4.4.2 Multipath Mitigation. Following multipath detection, the complex cepstrum and the adaptive weighted comb filter are used to remove the reflected signal in the cepstral domain. Then, a direct path estimate is obtained through an inverse complex cepstrum process. To determine the effectiveness of the complex cepstrum filtering process, two figures of merit are used. First, the mean squared amplitude error between the direct path estimate, and the actual direct path signal is calculated from the following formula

$$MSE = \frac{1}{N} \sum_{n=0}^{N-1} [x(n) - \hat{x}(n)]^2 \tag{79}$$

where $x(n)$ and $\hat{x}(n)$ are the n^{th} data points of the direct path signal and the direct path estimate, respectively.

The second figure of merit is the non-coherent delay lock loop tracking bias. This bias is calculated from the S-curve presented in Chapter III. The tracking bias is defined to be the magnitude of the delay or advance of the recovered signal S-curve relative to the direct path S-curve. The advance or delay is measured at the point where the S-curves are equal to zero. This is illustrated in Figure 37 which shows a typical, noise-free S-curve plot with the appropriate tracking biases noted on the figure.

4.4.2.1 Mean Squared Error Analysis. Figures 38 through 41 show the average mean squared amplitude error in the direct path estimate for multipath amplitudes of 0.2, 0.4, 0.6, and 0.8, and multipath delays from 0.1 chip to 1.5 chips, both relative to the direct path signal. For each pair of multipath amplitudes and delays, 10 trials are run at each of three different SNR levels, 1000 dB, 250 dB, 223 dB; the figures show the average values. The SNRs are chosen to collect data well above, slightly above, and near the threshold previously discussed. This gives an idea of how sensitive the complex cepstrum filtering process is to input SNR, assuming the SNR is above the threshold.

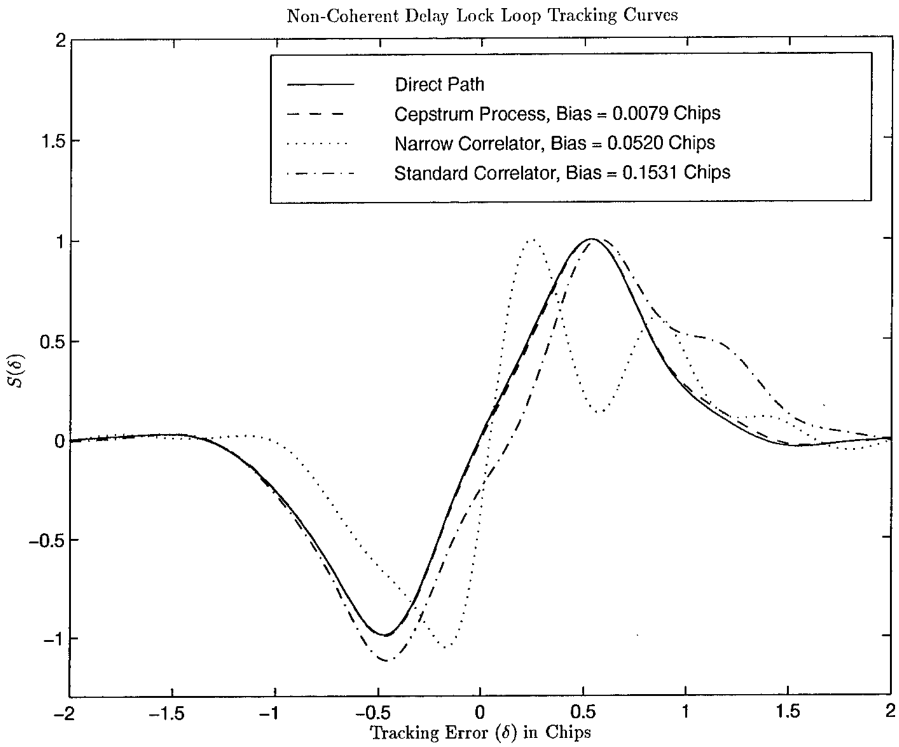

Figure 37 Non-Coherent Delay Lock Loop Tracking Curves, Multipath Normalized Amplitude = 0.6, Multipath Normalized Delay = 0.65 Chips

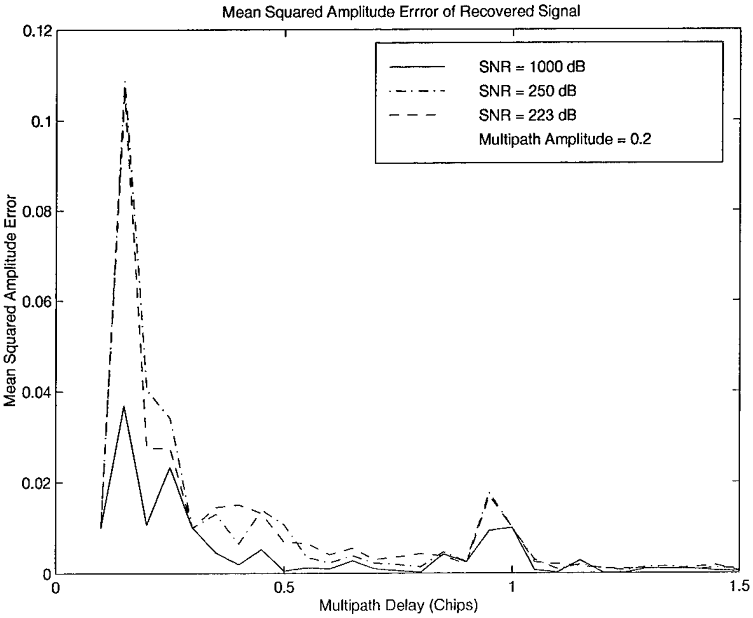

Figure 38 Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.2

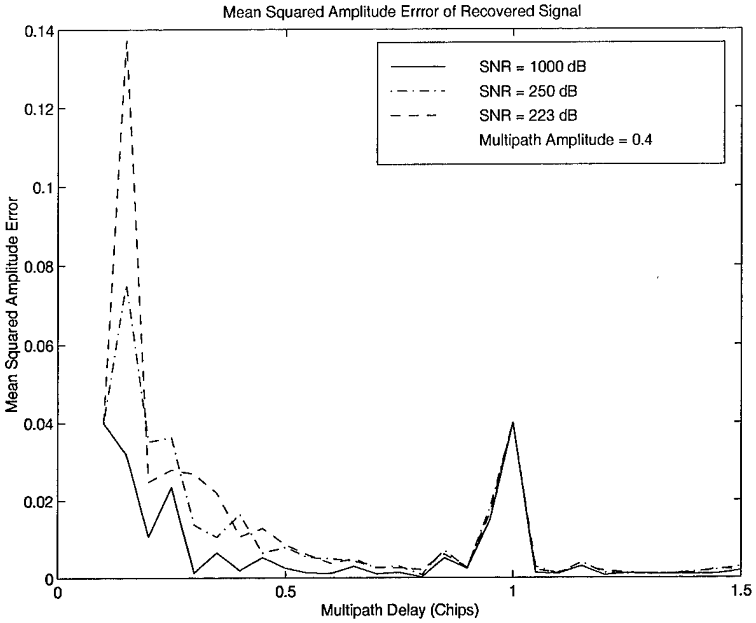

Figure 39 Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.4

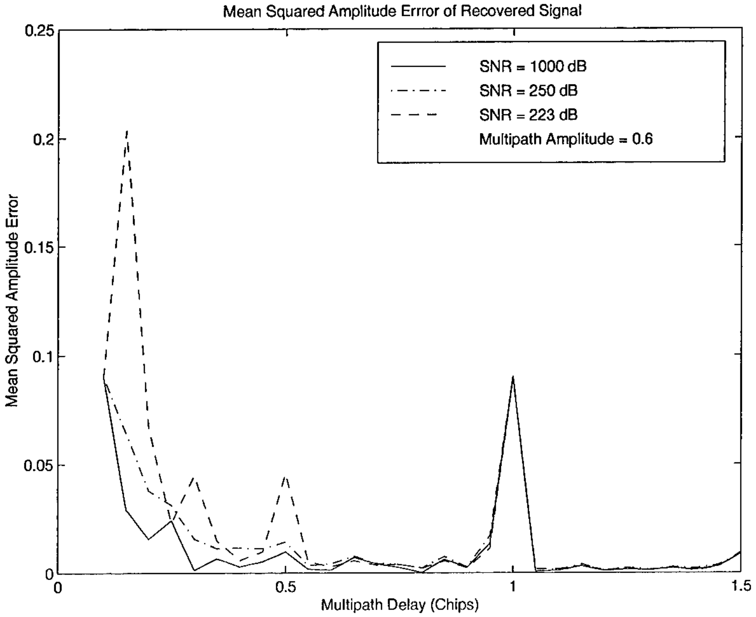

Figure 40 Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.6

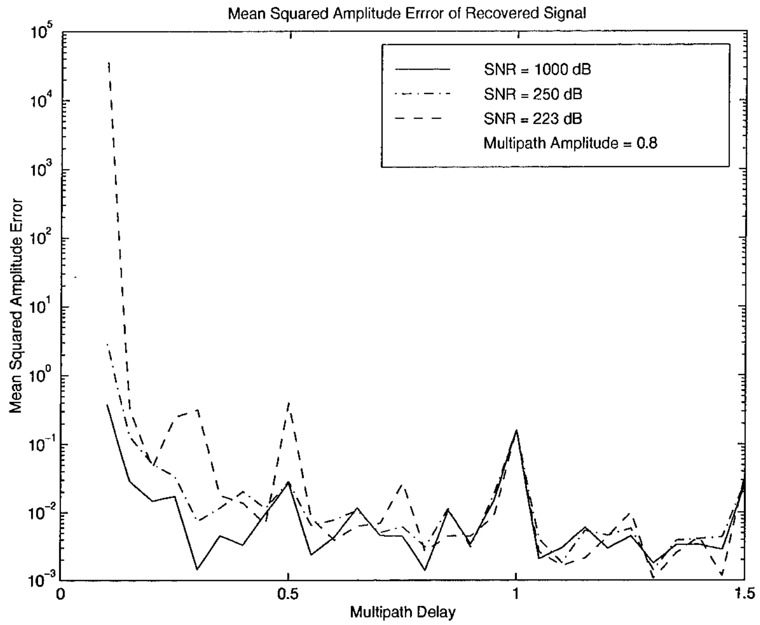

Figure 41 Direct Path Estimate Mean Squared Error For Multipath Amplitude = 0.8

Examination of Figures 38 through 41 reveals that the mean squared error is reasonably insensitive to SNR, assuming the threshold condition is met. In Figure 41, a condition where the SNR threshold is not met is observed. In this plot, the amplitude MSE for a multipath signal delayed 0.1 chip relative to the direct path, with an input SNR of 223 dB is extremely large. This is a direct result of the SNR threshold not being met for that pair of multipath parameters. In the next subsection, it will be seen that this large mean squared amplitude error in the estimated direct path signal amplitude leads to a large tracking bias. This figure also shows that 223 dB is near the SNR threshold also for a multipath delay of 0.15 chips. Other than those two data points, the mean squared error performance is adequate for all other delays at all three SNRs.

4.4.2.2 Tracking Bias Analysis. The second figure of merit for characterizing the cepstrum multipath mitigation process is the tracking bias. To perform this analysis, simulations are run for multipath amplitudes of 0.2, 0.4, 0.6. and 0.8 at SNRs of 1000 dB, 250 dB, and 223 dB. For each multipath amplitude and SNR, the multipath delay is varied from 0.1 to 1.5 chips. Again, 10 trials are run for each set of multipath parameters, and the data presented is the average for those 10 experiments. To characterize the cepstrum performance, the tracking bias of the cepstrum process is calculated and compared to the tracking bias for a narrow correlator non-coherent delay lock loop employing 0.1 chip correlator spacing. This comparison is performed to characterize the cepstrum process in relation to a current, commercially available, method of multipath mitigation. The performance of the cepstrum process is compared to that of the narrow correlator in Figures 42 through 53.

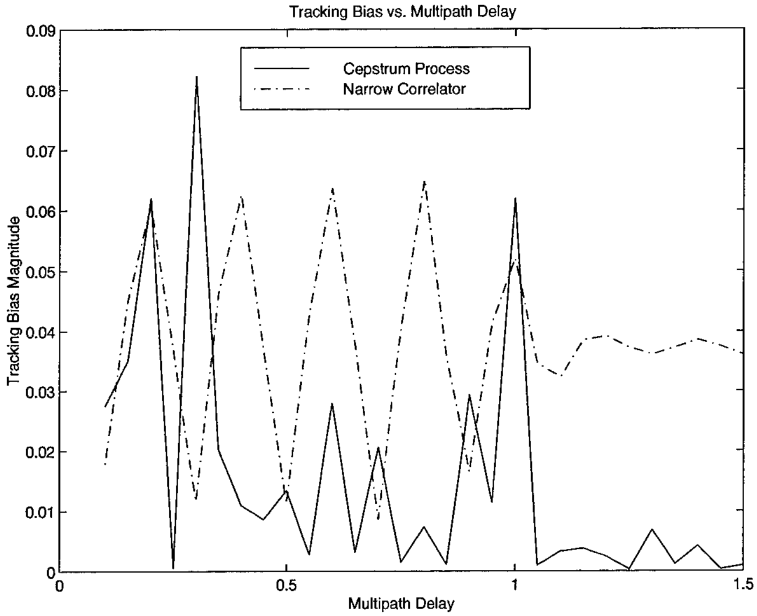

Figure 42 Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.2

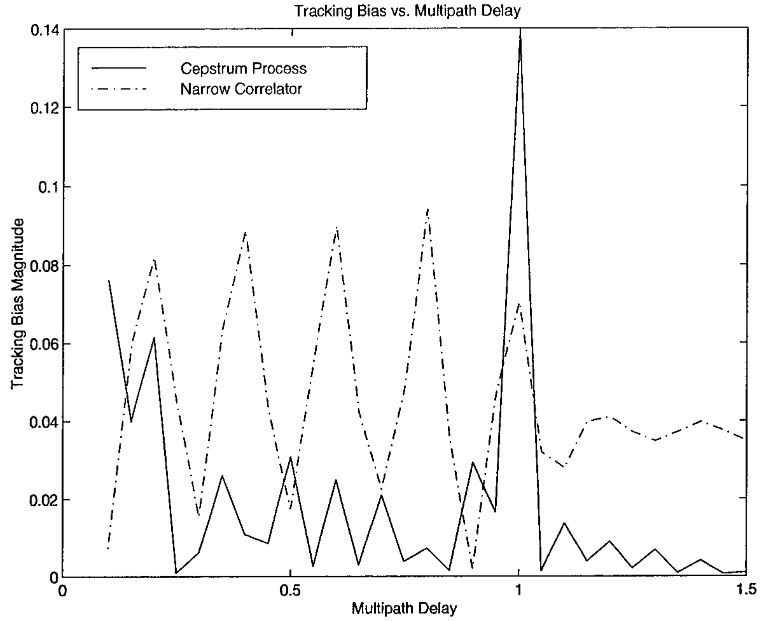

Figure 43 Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.4

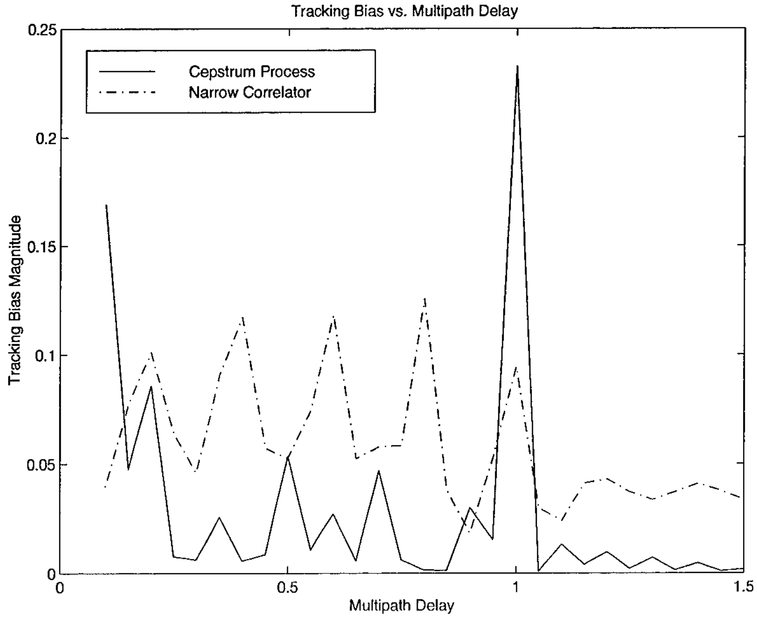

Figure 44 Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.6

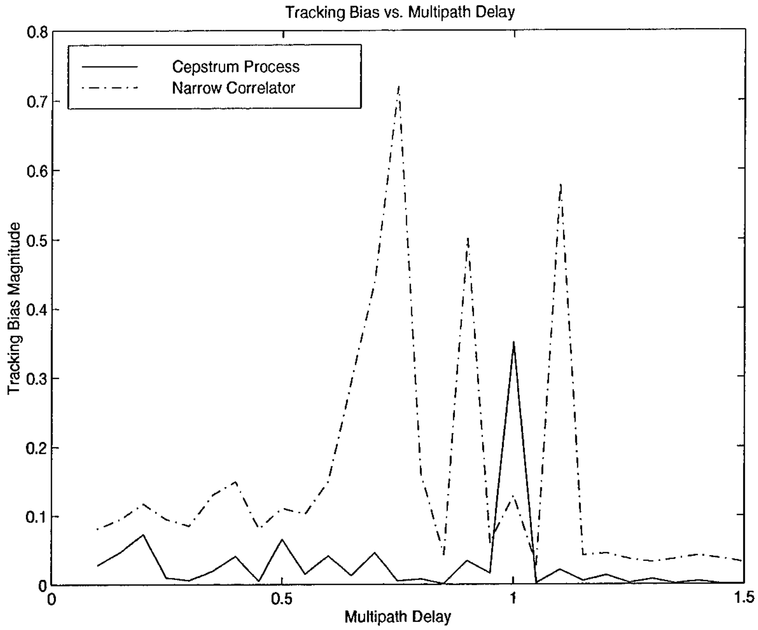

Figure 45 Tracking Bias Comparison for SNR = 1000 dB and Multipath Amplitude = 0.8

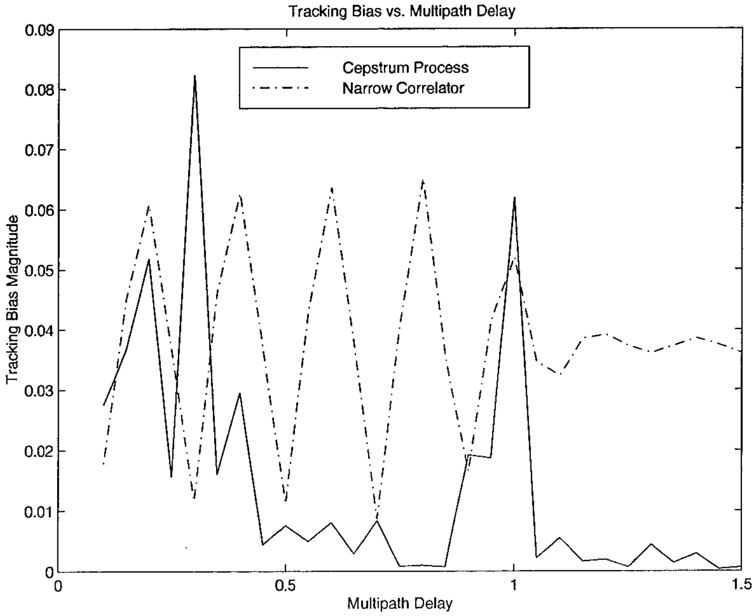

Figure 46 Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.2

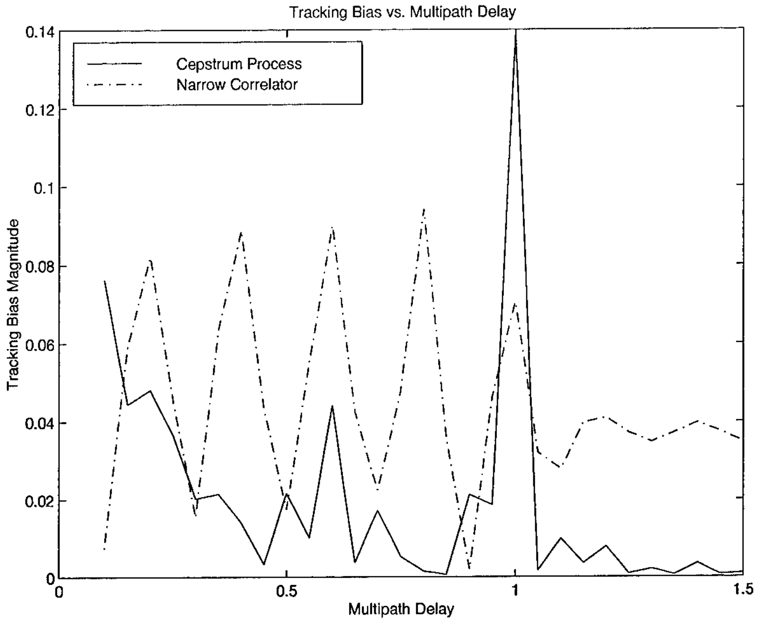

Figure 47 Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.4

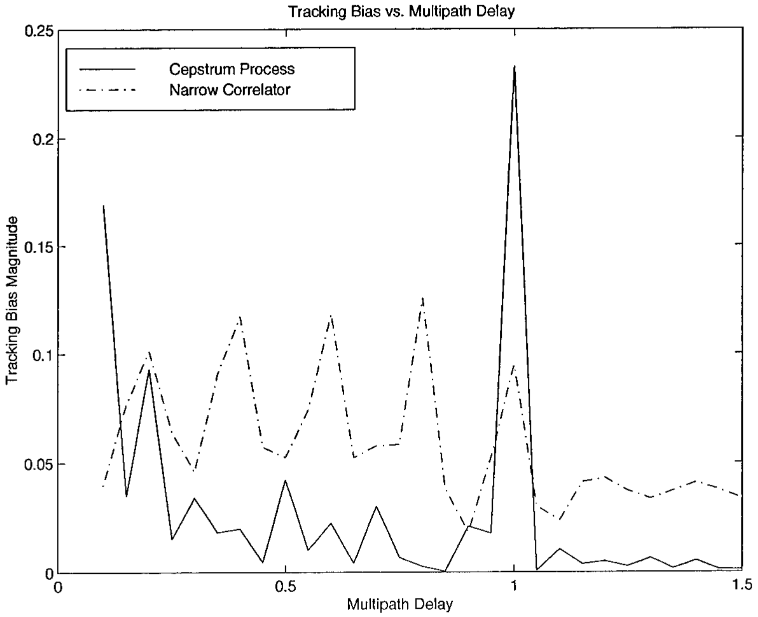

Figure 48 Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.6

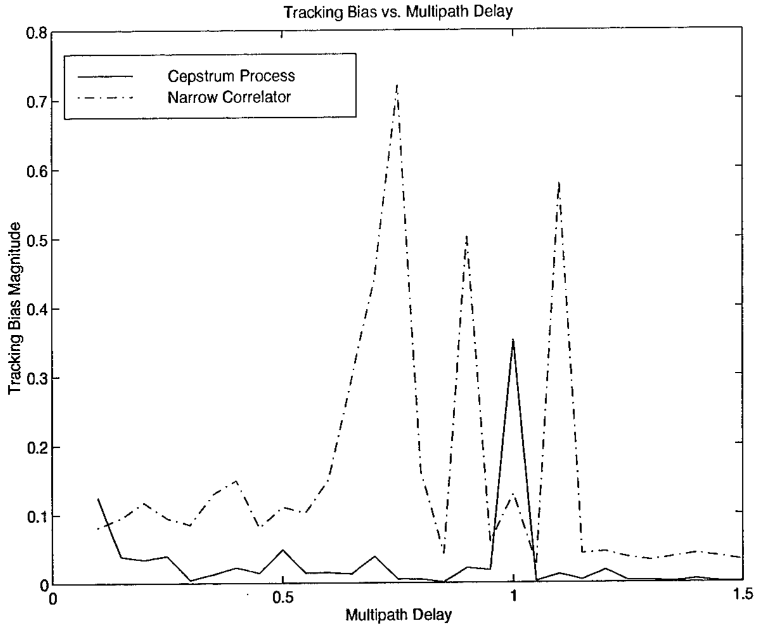

Figure 49 Tracking Bias Comparison for SNR = 250 dB and Multipath Amplitude = 0.8

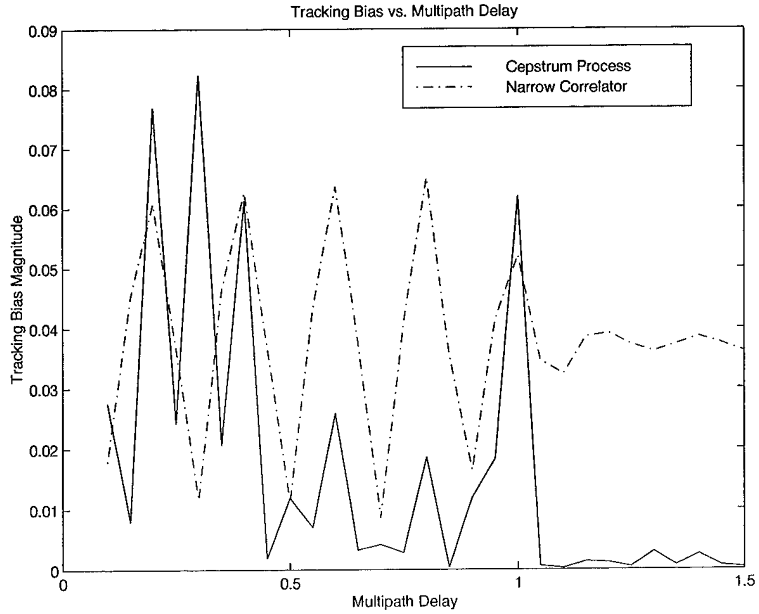

Figure 50 Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.2

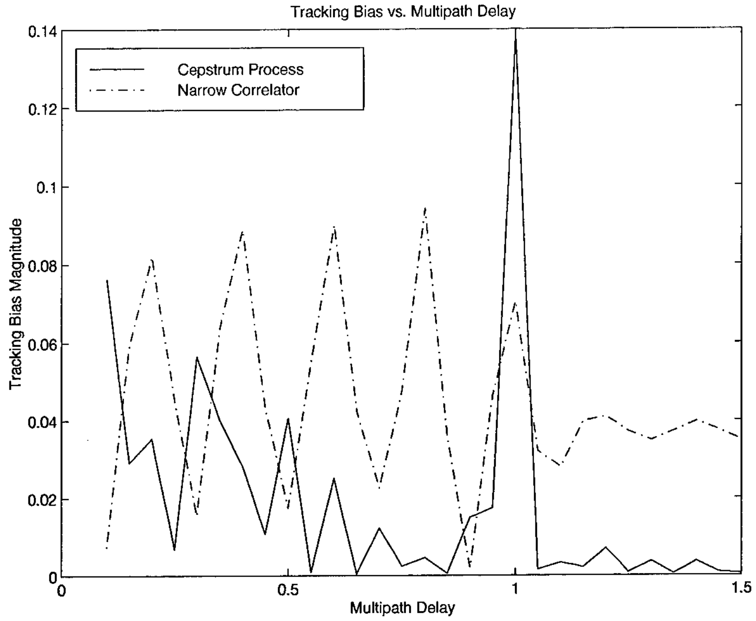

Figure 51 Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.4

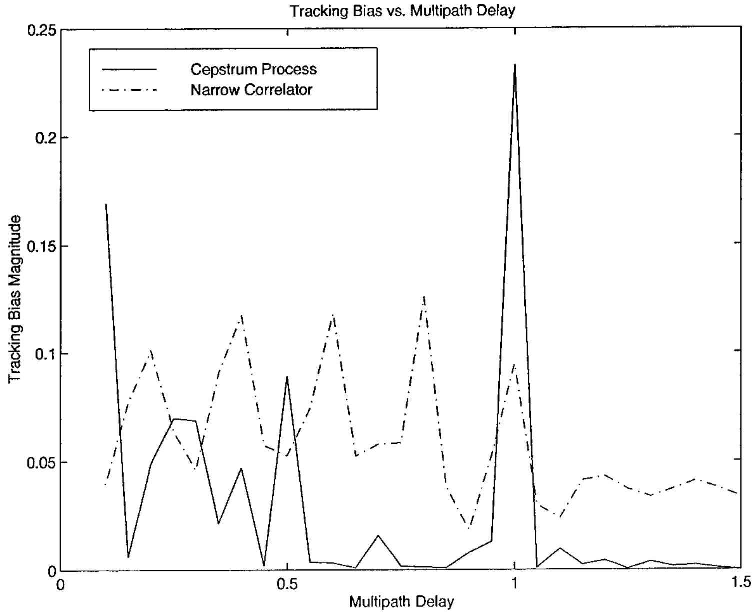

Figure 52 Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.6

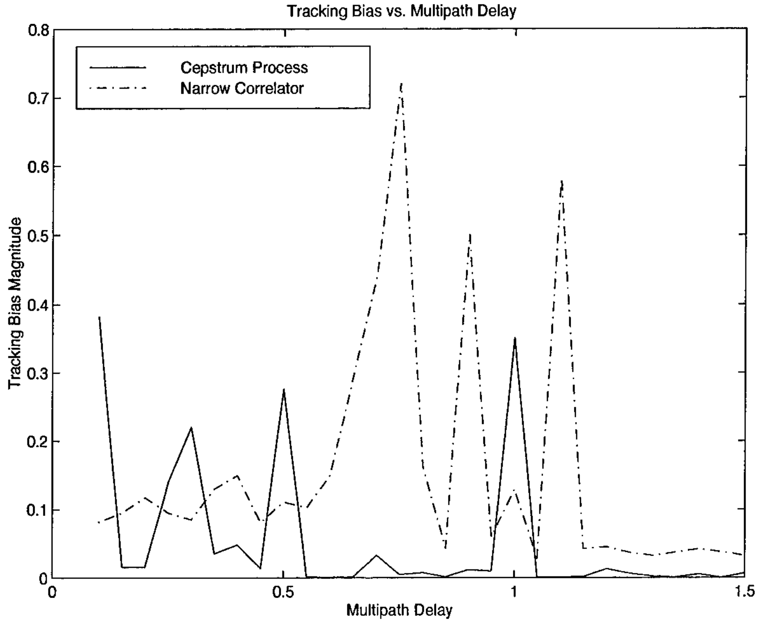

Figure 53 Tracking Bias Comparison for SNR = 223 dB and Multipath Amplitude = 0.8

Examination of Figures 42 through 53 reveals that the cepstrum process generally exhibits lower tracking bias than the narrow correlator, when the SNR threshold requirement is met. Exceptions to this general rule occur when the multipath delay is small and when it equals 1 chip. For small multipath delay, the adaptive comb filter must filter the complex cepstrum at more data points, introducing more opportunity for degradation in the recovered signal. For instance, if the multipath delay is 0.1 chip, the complex cepstrum must be filtered at every integer multiple of 0.1 chip. If, on the other hand the delay is 0.5 chip, it will be filtered at every integer multiple of 0.5 chip. The first example will be filtered at 5 times as many quefrencies as the second, introducing more opportunities for filtering error in the recovered signal. Additionally, it has been seen that, at small multipath delays, the detector frequently misdetects the multipath delay. This leads to the recovered direct path estimate more closely resembling the composite multipath signal rather than the direct path signal. However, all current multipath mitigation techniques exhibit poor performance for small reflection delays.

As previously mentioned, the second exception occurs when the reflection delay is 1.0 chip. Due to the importance of the cepstral peaks at each integer multiple of 1.0 chip, as discussed previously, the adaptive weighted comb filter is designed not to filter the cepstrum at multiples of the chip interval. Therefore, by design, if the multipath signal is delayed exactly 1.0 chip, relative to the direct path, the cepstrum process performs no filtering, and the original composite multipath signal is sent to the tracking loop. This weakness degrades performance, compared to the narrow correlator, in the one case where the multipath delay is 1.0 chip; however, it ensures better performance at all other multipath delays.

Finally, examination of Figure 53 reveals that the maximum tracking bias for the cepstrum process, when the input SNR is 223 dB and the multipath amplitude is 0.8, occurs at 0.1 chip multipath delay. Recall that the mean squared error of the recovered signal in that case is large due to the threshold effect. The poor quality of the recovered signal at this point leads to poor tracking performance as expected. Simulations reveal that, as the SNR is decreased further, tracking performance degrades and tracking cannot be accomplished.

4.5 Summary

In this chapter, data is presented from simulations characterizing the operation of the cepstrum process for multipath detection and mitigation. Simulations are run for three SNRs, 1000 dB, 250 dB, and 223 dB. At each SNR, a variety of multipath parameters are used to verify operation over a wide range of multipath scenarios. For each pair of multipath parameters and each SNR, 10 trials with independent Gaussian noise are accomplished. The plots in this chapter present the average values from these trials for each pair of multipath parameters. It should be noted that the variance in the parameters over the trials is extremely small and therefore is not presented here.

Table 2 Average Data Summary For 1000 dB SNR

Parameter	**Multipath Amplitude**			
	0.2	0.4	0.6	0.8
Mean Squared Error	0.0050	0.0072	0.0117	0.0262
Cepstrum Bias (chips)	0.0156	0.0191	0.0287	0.0262
Narrow Correlator Bias (chips)	0.0379	0.0445	0.0584	0.1540
Standard Correlator Bias (chips)	0.0322	0.0713	0.1183	0.2648

Table 3 Average Data Summary For 250 dB SNR

Parameter	**Multipath Amplitude**			
	0.2	0.4	0.6	0.8
Mean Squared Error	0.0107	0.0120	0.0159	0.1193
Cepstrum Bias (chips)	0.0145	0.0200	0.0282	0.0304
Narrow Correlator Bias (chips)	0.0379	0.0445	0.0584	0.1540
Standard Correlator Bias (chips)	0.0322	0.0713	0.1183	0.2648

A summary of the average performance, taken over all the multipath delays, for each SNR and multipath amplitude is shown in Tables 2 through 4. In addition to the narrow correlator data presented in the plots of this chapter, the tables also present data collected for a standard correlator using 1.0 chip correlator spacing. It is seen in the tables that the cepstrum process works

Table 4 Average Data Summary For 223 dB SNR

Parameter	**Multipath Amplitude**			
	0.2	0.4	0.6	0.8
Mean Squared Error	0.0105	0.0143	0.0229	0.0577 (edited)
Cepstrum Bias (chips)	0.0165	0.0195	0.0286	0.0556
Narrow Correlator Bias (chips)	0.0379	0.0445	0.0584	0.1540
Standard Correlator Bias (chips)	0.0322	0.0713	0.1183	0.2648

well over a wide variety of multipath scenarios using very high input SNRs. The "edited" line in Table 4 presents the average mean squared error with the one outlying data point (shown in Figure 41) removed.

V. Conclusion

This thesis seeks to provide a new method for GPS multipath mitigation. It proposes use of the cepstrum to detect and remove multipath effects prior to code tracking by a non-coherent delay lock loop. In Chapter II, the theory of the cepstrum is presented. Chapter III discusses the signal model, the cepstrum process, and the non-coherent delay lock loop. Chapter IV presents the results of simulations using the cepstrum for multipath mitigation.

Recall from Chapter I that the objectives of this thesis are:

1. Develop a complex cepstrum filtering technique for GPS multipath interference removal.
2. Modify a standard delay lock loop by adding the filtering to the loop input.
3. Characterize the multipath induced tracking error for the modified delay lock loop in a noiseless environment.
4. Characterize the multipath induced tracking error for the modified delay lock loop operating with typical GPS SNRs.
5. Compare and contrast the complex cepstrum receiver performance to that of a narrow correlator receiver.

These objectives were met through the theory of Chapters II and III, and demonstrated through the simulations of Chapter IV. Based on the results of Chapter IV, several conclusions can be made.

1. The power cepstrum can be used to detect multipath interference prior to GPS code tracking.
2. The complex cepstrum can be used to remove multipath interference prior to GPS code tracking.

3. Both the complex and power cepstra are sensitive to input SNR, with the complex cepstrum being extremely sensitive to input noise.
4. Time domain filtering to improve input SNR causes serious degradations in the cepstral domain.
5. Due to the noise sensitivity of the complex cepstrum, this technique is not viable at this time for use in actual GPS receivers where the input SNR is typically -14.9 dB.

Although the noise sensitivity of the complex cepstrum makes it unusable for GPS multipath mitigation, this research shows that the complex cepstrum is a viable technique for multipath mitigation in other circumstances. This technique may be applicable to other communication systems with different input signal structures and higher SNRs. One example of such a system may be overseas telephony where users often hear echoes while talking.

5.1 Recommendations for Future Research

Despite the fact that these cepstrum techniques do not appear viable for GPS multipath mitigation at this time, further research is warranted to investigate possible improvements to make the cepstrum a viable tool for GPS applications. This research includes the following:

1. Investigate cepstral processing for multipath mitigation within or following a non-coherent or, perhaps more promisingly, a coherent delay lock loop, rather than strictly prior to a NCDLL. Correlation prior to cepstral processing may lead to the signal to noise improvements necessary to make this technique applicable for GPS signals. A post delay lock loop cepstral algorithm has the advantage of greatly enhanced SNR at the expense of a distorted measured autocorrelation function (via a bank of evenly spaced correlators as in the MEDLL or MRDLL designs). Cepstral techniques could then be used to deconvolve the primary path autocorrelation function from the contribution of any component reflections.

2. Develop a new multipath signal model that accounts for the time domain convolution of an input filter impulse response. Characterize the filter influence in the cepstral domain, and develop methods of detecting and removing multipath despite the effects of the input filter.
3. Develop methods of cepstral domain filtering to minimize input noise effects prior to multipath detection and mitigation.
4. Using the theory of Chapter II, extend this research to include multiple reflections.
5. Improve the signal model to include effects of the carrier tracking loop, and characterize the performance of the cepstrum process on this signal.
6. Characterize the cepstrum process when doppler effects are taken into account.
7. Characterize the cepstrum process for GPS carrier tracking multipath mitigation.

5.2 Final Conclusion

This work shows the cepstrum is viable for multipath mitigation under ideal, noiseless conditions. The simulations show that the cepstrum process requires an extremely high input SNR. Further research is necessary to improve noise performance before this will be a viable technique for GPS code tracking multipath mitigation.

Appendix A. Matlab Function Files

A.1 Overview

The Appendix contains copies of all Matlab (version 5) function files used for simulation of multipath mitigation via cepstral techniques. Each file is a separate section of the Appendix. A short introduction to each section is given, followed by a listing of the function file.

A.2 Thesis_Simulator Function File

The function file *thesis_simulator* is the primary file used for simulation of cepstral processing for multipath mitigation. This file creates the composite multipath signal, calculates the power and complex cepstra, calls the detector subroutine to detect the reflection delay, removes the multipath effects in the cepstral domain, and produces the direct path estimate through an inverse complex cepstrum operation. Additionally, this file calculates the mean squared amplitude error in the direct path estimate, compared to the actual direct path signal. Finally, it calculates the S-curves for the direct path signal, the direct path estimate, the narrow correlator NCDLL, and the standard correlator NCDLL. These S-curves are then used to calculate the tracking biases of the cepstrum process, the narrow correlator NCDLL, and the standard correlator NCDLL. The Matlab function file follows:

```
function [results]=thesis_simulator(am,tm,SNR)

% THESIS_SIMULATOR uses the cepstrum to detect and remove a multipath reflection from a
% composite signal.  The direct path signal is modeled as a cosine modulated by a BPSK
% spreading code.  The multipath reflection is modeled as an amplitude scaled and time
% delayed version of the direct path signal.  The composite signal is the sum of the
% direct path and reflection signals.
%
% Usage:
% results=thesis_simulator(am,tm,SNR)
%
% Where:
% am = multipath reflection normalized amplitude
% tm = multipath reflection normalized delay (in chips)
% SNR = the input signal to noise ratio in dB
%
```

```
% This file calls the following subroutines:
%
% expand, delay, fpeak, meanfilt, scurve, trackpoint
%
% This function and all subroutines were written by Chuck Ormsby
%
load cacode % Load pre-stored C/A code generated using the GPS toolbox for Matlab
fs=100; % fs is the sampling frequency (ie 100 samples/chip)
ca=[expand(ca,fs)]; % Produces 100 samples per chip
ca=[ca zeros(size(ca))]; % Zero pad the signal for cepstral processing
t=linspace(0,2*pi*1023*2,length(ca)); % Time scale for the carrier
carrier=cos(5*t);
y=ca.*carrier; % Direct path signal
n=(0:length(ca)-1)/100;

% Add a multipath signal
d=round(tm*fs); % Reflection delay  Note:  Round is necessary to make sure d is an integer.
a=am; % Reflection Amplitude

mp=a*delay(y,d); % Multipath reflection signal
cs=y+mp;  % Composite signal

% Convert the SNR in dB to a noise amplitude

na=exp(-SNR/20);

% Generate a Gaussian random sequence

randn('state',sum(100*clock)); % Reset the random number generator.
rv=na*randn(size(cs));

% Add the noise to the signal
cs=cs+rv;       % Noisy composite signal

% Caculate the power and complex cepstra
[CS,nd]=cceps(cs); % Complex cepstrum; nd is delay which must be removed later
RCS=rceps(cs);  % Power cepstrum

% Use the power cepstrum to detect the multipath delay
first_peak=fpeak(abs(RCS),.6); % Detect the multipath peak
detected_delay=(first_peak-1)/fs; % Detected delay in chips

% Filter the multipath out

max_multiplier=floor(length(CS)/first_peak);
multipliers=1:max_multiplier;
filter_points=first_peak*multipliers;

for index=1:length(filter_points)
    if rem(filter_points(index)-index+1,100)~=1 % Do not filter at the chip times
       CS=meanfilt(CS,filter_points(index)-index+1,2); % Filters out the delta functions
    end
end

% Recover the direct path estimate
ca1=icceps(CS,nd); % Inverse cepstrum calculation

% Calculate the mean squared amplitude error in the recovered signal
mse=mean((y-ca1).^2);  % MSE in the recovered signal

% Calculate the s-curves of the actual direct path, the direct path
% estimate, the narrow correlator receiver and the standard NCDLL

[sdp,t]=scurve(y,y,fs,1); % Direct path s-curve and tracking error scale (t)
sc=scurve(ca1,y,fs,1); % Cepstrum process s-curve
snc=scurve(cs,y,fs,.1); % Narrow correlator s-curve
sncdll=scurve(cs,y,fs,1); % Standard correlator s-curve

% Calculate the tracking biases for the direct path estimate, the
% narrow
% correlator receiver, and the standard NCDLL relative to the direct path signal

tpdp=trackpoint(sdp); % Find the zero crossing point of the direct path s-curve
tpcep=trackpoint(sc); % Find the zero crossing point of the cepstrum process s-curve
tpnc=trackpoint(snc); % Find the zero crossing point of the narrow correlator s-curve
tpncdll=trackpoint(sncdll); % Find the zero crossing point of the standard correlator s-curve
cep_bias=(tpcep-tpdp)/fs; % Calculate the cepstrum tracking bias in chips
nar_bias=(tpnc-tpdp)/fs; % Calculate the narrow correlator tracking bias in chips
stan_bias=(tpncdll-tpdp)/fs; % Calculate the standard correlator tracking bias in chips

% Report the results of the simulation.
results=[SNR am tm detected_delay mse cep_bias nar_bias stan_bias];
```

A.3 Expand Function File

Expand is the first function file called by *thesis_simulator*. *Expand* expands an input vector containing one sample per time unit, to one containing f_s samples per time unit, where f_s is the sampling frequency. For this specific application, *expand* takes an input vector of GPS C/A spreading code containing one sample per chip, and expands the vector to contain 100 samples per chip. The Matlab function file follows:

```
function yp=expand2(y,expan)

% EXPAND will expand a vector by an input factor.  This expansion is equivalent to producing
% more samples per time unit.
```

```
%
% ex.  y=[1 2 3 4]
%      yp=expand(y,2)
%      yp=[1 1 2 2 3 3 4 4]
%
% yp=expand(y,expansion factor)
%
% Written by Chuck Ormsby
% April 20, 1997
%

yp=zeros(1,length(y)*expan);
count=1;
count2=1;
while count2<=length(y)
   yp(count:count+expan-1)=y(count2)*ones(1,expan);
   count2=count2+1;
   count=count+expan;
end
```

A.4 Delay Function File

The function file *delay* adds a delay to the input vector by prepending zeros to the beginning of the vector. This function is used to delay the direct path signal vector for creation of the reflection signal vector. The Matlab function file follows:

```
function y=delay(x,s)

% DELAY prepends zeros to the beginning of the input vector.  This can be thought of as
% adding a time delay to the input vector.
%
% y=delay(x,s)
%
% y is the delayed vector, x is the input vector, s is the length of the delay and must
% be positive
%

left=zeros(1,s);
right=x(1:length(x)-s);
y=[left right];
```

A.5 Fpeak Function File

The function file *fpeak* detects the multipath reflection delay. This function compares each data point in the input vector to the two adjacent data points as described in Chapter III. When a delta function peak is detected, the function reports the associated delay as the reflection delay. The Matlab function file follows:

```
function first_peak=fpeak(x,multiple)

% FPEAK detects the first peak in a function.  It is designed to work with the CCAF.
%   A peak is declared if a point exceeds the points on either side of it by more than
%   an input multiple which is a percentage  (ie multiple = .5 means a point must be
%   2 times larger than the adjacent point).  By default, the first point cannot be a peak.
%
% first_peak = fpeak(x,multiple)
%
% Written By:
% Chuck Ormsby
% 27 Oct 97
%

stop=0;
index=2;
while stop==0
        if x(index)>=(1/multiple)*x(index-1) & x(index)>=(1/multiple)*x(index+1)
                if index~=101 % Do not declare an mp delay of 1 chip
                        first_peak=index;
                        stop=1;
                else
                        index=index+1;
                end
        else
                index=index+1;
        end
end
```

A.6 Meanfilt Function File

The *meanfilt* function file accomplishes the removal of the multipath reflection effects from the complex cepstrum. This function replaces the appropriate data points in the complex cepstrum with the average of the adjacent data points. The Matlab function file follows:

```
function xf=meanfilt(x,m,N)
```

```
% MEANFILT is a "mean" comb filter designed for use with the complex cepstrum.  This
% filter replaces element m of the vector, x, with the mean of the N surrounding points.
% N must be even.
%
% Ex: x=[5 2 3 4 5 6]
%
% xf=meanfilt(x,3,4)
% x(3)=(5+2+4+5)/4
% x(3)= 4
% xf=[5 2 4 4 5 6]
%
% Note: x and m must be specified.  If N is not entered, a default value of N=2 is used.
%
% Written by Chuck Ormsby
%
if nargin==2
   N=2;
end
% Test N
while rem(N,2)~=0
   N=input('N must be even.  Please enter a new N.   ');
end
x(m)=(sum(x(m-N/2:m-1))+sum(x(m+1:m+N/2)))/N;
xf=x;
```

A.7 Scurve Funtion File

Scurve calculates the S-Curve for an input sequence as defined in Chapter III. *Scurve* correlates an input signal with a locally generated signal, which is also an input to the function. This function allows for variable correlator spacing so that narrow or standard correlator NCDLL S-curves can be calculated using the same function file. The Matlab function file follows:

```
function [sc,t]=scurve(x,c,fs,s)

% SCURVE computes the tracking curve (s-curve) for a non-coherent delay lock loop where x
%   is the input signal, c is the correlation signal, fs is the sampling frequency,
%   and s is the correlator spacing.  If s is not specified, the default is 1 chip.
%   If two output variables are used, a time scale for the s-curve is also returned.
%
% [sc,t]=scurve(x,c,fs,s)
%
% Written by Chuck Ormsby
%           August 19, 1997
%
% This function calls the vector_shift subroutine:
```

```
%

% Note:  Signal names used in this file refer to the names in figure 4-9, pg 165 of
% Peterson, Ziemer, and Borth: Introduction to Spread Spectrum Communications
if nargin==3
   s=1;
end
if rem(s*fs,2)==0
   space=s*fs/2;
else
   space=(s*fs+1)/2;
end
% Test the length of the correlation signals
len=length(x);
if len>=50000
   trun=5e4;  % For very long sequences, the input signal must be truncated to decrease
              % run times
   else
      trun=length(x);
   end
end
x=x(1:trun);
c=c(1:trun);
% Produce the early and late codes
early=vector_shift(c,space);
late=vector_shift(c,-space);
% Produce the early and late correlator outputs
y1=xcorr(early,x);
y2=xcorr(late,x);
% Square the correlator outputs
z1=y1.^2;
z2=y2.^2;
% Low pass filter
wcut=0.035;
[b,a]=butter(5,wcut);
z1lp=filter(b,a,z1);
z2lp=filter(b,a,z2);
% Create the s-curve
sc=z1lp-z2lp;
sc=sc./max(sc); % Normalize the s-curve
t=linspace(-length(x),length(x),length(sc))/fs;
```

A.8 Vector_Shift Function File

The function file *vector_shift* produces the time shift necessary for the early and late locally generated codes in the NCDLL. This function assumes the input vector is one period of a periodic spreading code. The function then shifts the elements of the vector circularly left or right to create an advanced or delayed version of the input vector. The Matlab function file follows:

```
function yp=vector_shift2(y,s)

% VECTOR_SHIFT shifts the entries in the vector y by some advance or delay.  A delay is
%   entered as a negative number while an advance is entered as a positive number.
%
% yp=vector_shift(y,s)
%
% Note:  The vector y is assumed to be periodic, with the elements of y comprising an integer
%   number of periods.
%
s=mod(s,length(y));
if s>=0
   right=y(1:abs(s));
   left=y(abs(s)+1:length(y));
else
   right=y(1:length(y)+s);
   left=y(length(y)+s+1:length(y));
end
yp=[left right];
```

A.9 Trackpoint Function File

The *trackpoint* function file searches for the tracking point in a vector representing the NCDLL S-curve. This function searches for the point where the S-curve crosses the $S(\delta) = 0$ axis between the minimum and maximum points of the S-curve. The function then reports the index of the data element of the zero crossing which can be related to a time delay through knowledge of the sampling frequency. The Matlab function file follows:

```
function z=trackpoint(x)

% Trackpoint searches for the tracking point of an input S-curve vector.  The
% tracking point is the point where the S-curve crosses the S=0 axis between
% the minimum and maximum values of the S-curve.  If the vector does not contain
% an element which is identically zero, a linear interpolation between the two
% nearest elements is performed.
%
% z = trackpoint(x)
%
% Written by Chuck Ormsby
%
% This function calls the subroutines:
```

```
% maxfind.m, minfind.m, and interpolate.m
%
max_index=maxfind(x);
min_index=minfind(x);
if min_index>max_index
        start=max_index;
        stop=min_index;
        count=1;
        for index=start:stop
                if x(index-1)>=0 & x(index)<=0
                        a(count,:)=[index-1 index];
                        count=count+1;
                end
        end
else
        start=min_index;
        stop=max_index;
        count=1;
        for index=start:stop
                if x(index-1)<=0 & x(index)>=0
                a(count,:)=[index-1 index];
                count=count+1;
                end
        end
end
[m,b]=interpolate(a(1),x(a(1)),a(2),x(a(2)));
z=-b/m;
```

A.10 Maxfind and Minfind Function Files

The *maxfind* and *minfind* function files search for the maximum and minimum points, respectively, in an input vector. These functions report the index of the maximum and minimum values of the input vectors. The Matlab function files follow:

```
function index=maxfind(x)

% MAXFIND returns the index associated with the maximum value of a vector.
%
% index=maxfind(x)
%
% Written by Chuck Ormsby
% July 24, 1997
%
maxi=max(x);
for count=1:length(x)
   if x(count)==maxi;
      index=count;
   end
end


function index=minfind(x)

% MINFIND returns the index associated with the minimum value of a vector.
%
% index=minfind(x)
%
% Written by Chuck Ormsby
% October 1, 1997
%

mini=min(x);
for count=1:length(x)
   if x(count)==mini;
      index=count;
   end
end
```

A.11 Interpolate Function File

The *interpolate* function file is used in conjunction with the *trackpoint* function to determine the tracking point of an input S-curve vector. *Interpolate* returns the slope and y-intercept of a line joining two input data points. This file is used by the *trackpoint* function to calculate the slope and y-intercept of a line joining two adjacent data points, one negative and one positive, in the S-curve vector. From knowledge of the slope and y-intercept of this line, the exact zero crossing point can be calculated. This linear interpolation is justified because, as shown in Chapter III, for small tracking errors, the S-curve is approximately linear. The Matlab function file follows:

```
function [m,b]=interpolate(x1,y1,x2,y2)

% INTERPOLATE returns the slope and y-intercept for a line connecting the points (x1,y1) and
% (x2,y2).
%
% [m b]=interpolate(x1,y1,x2,y2)
%
m=(y2-y1)/(x2-x1);
b=y1-m*x1;
```

A.12 Summary

This Appendix presents the Matlab function files used for simulation of multipath mitigation using cepstral techniques. All function files used for this thesis are included in this Appendix. Any functions not specifically included are built-in Matlab (version 5) functions. The intent of including the function files is to give the reader a better understanding of the function of the cepstral process for multipath mitigation. Additionally, by including these function files, a reader familiar with Matlab should be able to reproduce the work of this thesis.

Bibliography

1. Bogert, B. P., et al. *Time Series Analysis*, chapter 15, 209–243. Wiley, 1963.
2. Childers, Donald G. and Robert C. Kemerait. "Signal Detection and Extraction by Cepstrum Techniques," *IEEE Transactions on Information Theory*, *IT-18*(6):745–759 (November 1972).
3. Childers, Donald G., et al. "The Cepstrum: A Guide to Processing," *Proceedings of the IEEE*, *65*(10):1428–1443 (October 1977).
4. Fisher, Stephen D. *Complex Variables*, chapter 1, 45–48. Wadsworth and Brooks/Cole, 1990.
5. Laxton, Mark L. *Analysis and Simulation of a New Code Tracking Loops for GPS Multipath Mitigation*. MS thesis, Air Force Institute of Technology, 1996.
6. Lee, James K., et al. "The Complex Cepstrum Applied to Two-Dimensional Images," *Pattern Recognition*, *26*(10):1579–1592 (1993).
7. Oppenheim, Alan V., et al. "Nonlinear Filtering of Multiplied and Convolved Signals," *Proceedings of the IEEE*, *56*(8):1264–1291 (August 1968).
8. Oppenheim, Alan V. and Ronald W. Schafer. *Discrete-Time Signal Processing*, chapter 12, 768–825. Prentice Hall, Inc., 1989.
9. Parkinson, Bradford W. and James J. Spilker Jr., editors. *Global Positioning System: Theory and Applications*, *1*, chapter 3, 118. American Institute of Aeronautics and Astronautics, Inc., 1996.
10. Peterson, Roger L., et al. *Introduction to Spread Spectrum Communications*, chapter 3, 135–138. Prentice Hall, 1995.
11. Sklar, Bernard. *Digital Communications Fundamentals and Applications*, chapter 10, 548 – 549. Prentice Hall, Inc., 1988.
12. van Dierendonck, A. J., et al. "Theory and Performance of Narrow Correlator Spacing in a GPS Receiver," *Navigation: Journal of the Institute of Navigation*, *39*(3):265–283 (Fall 1992).
13. van Nee, Richard D. J., et al. "The Multipath Estimating Delay Lock Loop: Approaching Theoretical Accuracy Limits." *Proceedings of 1994 IEEE Position, Location, and Navigation Symposium - PLANS 94*. 246–250. April 1994.
14. Weill, Lawrence R. "GPS Multipath Mitigation by Means of Correlator Reference Waveform Design." *Proceedings of the National Technical Meeting of the Institute of Navigation*. 197–206. January 1997.

Vita

Charles D. Ormsby was born on May 28, 1970 in Terre Haute, IN. He was commissioned as a U.S. Air Force officer on May 22, 1992, and received a B.S. degree in Electrical Engineering from Rose-Hulman Institute of Technology on May 23, 1992. Since receiving his commission, he has worked as a foreign microelectronics materials analyst and a foreign materiel exploitation engineer at the National Air Intelligence Center, Wright-Patterson AFB, OH. Immediately following completion of his M.S. degree in Electrical Engineering from the Air Force Institute of Technology, he will relocate to Holloman AFB, NM to perform GPS system testing.

Charles Ormsby is a member of Pi Kappa Alpha, Tau Beta Pi, and Eta Kappa Nu.

Permanent address: 6241 North Clinton Street
Terre Haute, IN 47805

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
December 1997

3. REPORT TYPE AND DATES COVERED
Master's Thesis

4. TITLE AND SUBTITLE
Cepstral Processing For GPS Multipath Detection and Mitigation

5. FUNDING NUMBERS

6. AUTHOR(S)
Charles D. Ormsby

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GE/ENG/97D-19

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
WL/AAMW
2241 Avionics Cirlce
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This work presents a novel approach to code phase multipath mitigation for Global Positioning System (GPS) receivers. It uses the power and complex cepstra for multipath detection and mitigation prior to code phase tracking by a standard non-coherent delay lock loop. Cepstral theory is presented to demonstrate how multipath reflection delays can be detected through the use of the power cepstrum. Filtering can then be performed on the complex cepstrum to remove multipath effects in the cepstral domain. Finally, an inverse complex cepstrum is calculated yielding a theoretically multipath free direct path estimate in the time domain. Simulations are presented to verify the applicability of cepstral techniques to the problem of GPS multipath mitigation. Results show that, under noiseless conditions, cepstral processing prior to code tracking by a standard non-coherent delay lock loop leads to lower code tracking biases than direct tracking of the composite multipath signal by a narrow correlator receiver. Finally, this work shows that cepstral processing is highly sensitive to additive white Gaussian noise effects, leading to the conclusion that methods of limiting noise effects must be developed before this technique will be applicable in actual GPS receivers.

14. SUBJECT TERMS
Global Positioning System, GPS, Coarse/Acquisition Code, C/A Code, Multipath Mitigation, Multipath Detection, Cepstrum, Power Cepstrum, Complex Cepstrum

15. NUMBER OF PAGES
115

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
UNCLASSIFIED

18. SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. SECURITY CLASSIFICATION OF ABSTRACT
UNCLASSIFIED

20. LIMITATION OF ABSTRACT
UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)